STUDY REPORT PHASE II

CLOSED-LOOP PROCESSING OF TOPSIDE IONOSPHERIC SOUNDER DATA

Contract No. NAS2-4273

Prepared for

NASA/ARC
Moffett Field, Calif.

Prepared by

ASTRODATA, Inc.
240 E. Palais Rd.
Anaheim, Calif.

R. Glen Madsen

R. Glen Madsen
Study Project Manager

February 20, 1970

UNCLASSIFIED

TABLE OF CONTENTS

TABLE OF CONTENTS (cont)

LIST OF ILLUSTRATIONS

REFERENCES

[1] L. Colin, K. L. Chan, R. G. Madsen, "FILMCLIP, Closed-Loop Ionogram Processor for the Analysis of Topside Sounder Ionogram Films," to be published, 1970.

[2] W. Cooper, "Scaling Converter Format for Closed-Loop-Ionogram-Processor (CLIP)", Informatics Project 780 - Technical Note 101, May 3, 1968.

[3] J. G. K. Lee, "Computations of Satellite Position and Gyro-frequency for the Closed-Loop-Ionogram-Processor (CLIP)", Informatics Project 780 - Technical Note 103, May 14, 1968.

[4] J. G. K. Lee, "Computations of Satellite Position and Gyro-frequency for the Closed Loop Ionogram Processor", Informatics Project 780 - Technical Note 104, May 20, 1968.

[5] J. G. K. Lee, "Computation of Satellite Position and Gyro-frequency for the Closed-Loop-Ionogram-Processor (CLIP)", Informatics Project 780 - Technical Note 105, June 3, 1968.

[6] J. G. K. Lee, W. R. Cooper, G. Slike, R. G. Madsen, "Description of Programmed Function Keys for Closed-Loop-Ionogram-Processor (CLIP)", Informatics Project 780 - Technical Note 102, May 24, 1968.

[7] G. Slike, "FILMCLIP 1800 Program Documentation," Informatics TR-1064-5-3, January 23, 1969.

[8] W. R. Cooper, "FILMCLIP Summary," Informatics TR-1064-5-1, February 5, 1969.

[9] G. Slike, J. G. K. Lee, "FILMCLIP User's Guide," Informatics TR-1064-5-4, February 5, 1969.

[10] J. G. K. Lee, "FILMCLIP 360 Program Documentation," Informatics TR-1064-5-2, May 29, 1969.

[11] W. R. Cooper, G. Slike, J. G. K. Lee, "CLIP-An Interactive Multi-Computer System for Processing Ionospheric Data," AIAA Paper No. 69-952, AIAA Aerospace Computer Systems Conference, Los Angeles, Calif., September 8-10, 1969.

[12] Astrodata, Inc., "Scaling Converter," Instruction Manual, Project 938400, January 30, 1970.

[13] J. E. Jackson, "The Reduction of Topside Ionograms to Electron-Density Profiles," Proc. IEEE, June 1969.

[14] J. E. Jackson, "Comparison Between Topside and Ground-Based Soundings," Proc. IEEE, June 1969.

[15] G. E. K. Lockwood, "A Modified Iteration Technique for use in Computing Electron Density Profiles from Topside Ionograms," Accepted for publication in Radio Science.

[16] R. G. Madsen, A. F. Scardina, N. L. Dawirs, "Engineering Study and Preliminary Design of Instrumentation for the Scaling and Editing of Ionograph Data," Astrodata, Inc. Study Report, Phase I Part I, Contract No. NAS2-4273, November 1967.

[17] J. G. K. Lee, "Reduction of Alouette II Topside Ionograms to Electron Density Profiles," Program Documentation for NASA/ARC, 1967.

[18] J. G. K. Lee, "Reduction of Electron Density Profiles to Alouette II Topside Ionograms," Program Documentation for NASA/ARC, 1967.

[19] R. C. K. Lee, "Optimal Estimation, Identification, and Control," Research Monograph No. 28, The M.I.T. Press, 1964.

[20] D. D. McCracken, "Fortran with Engineering Applications," Wiley, 1967.

The following references are in the Appendix:

[21] J. E. Kinkel, "Some Proposed Methods for Reduction of Topside Ionograms to Electron Density Profiles," Astrodata Project 938400 - Technical Report 1, September 25, 1969.

[22] J. F. Kinkel, "A Weighted Least Squares Approximation Method," Astrodata Project 938400 - Technical Report 2, November 21, 1969.

[23] J. F. Kinkel, "Inverse Mapping to Specified f," Astrodata Project 938400 - Technical Report 3, November 24, 1969.

[24] J. F. Kinkel, "Simplification of Matrix Inversion Problem," Astrodata Project 938400 - Technical Report 4, December 1, 1969.

[25] J. F. Kinkel, "Time Skew Problem," Astrodata Project 938400 - Technical Report 5, December 12, 1969.

[26] J. F. Kinkel, "Optimal Step Size by One Dimensional Search," Astrodata Project 938400 - Technical Report 6, December 30, 1969.

[27] J. F. Kinkel, "Horizontal Processing," Astrodata Project 938400 - Technical Report 7, January 30, 1970.

[28] J. F. Kinkel, "Computation of the Pseudoinverse," Astrodata Project 938400 - Technical Report 8, February 16, 1970.

INTRODUCTION

The Phase II Study was undertaken to investigate the closed-loop data reduction techniques described in the first Study Report[16] and to investigate some new ideas that had evolved since that report was written. With the availability of the computer hardware at NASA/ARC, it required only the addition of the interface hardware between the Oscar-F Data Reader and the Computer, plus the associated software, to implement what was initially called the Phase II simulator. This simulator has now become the FILMCLIP System which has provided NASA/ARC with closed-loop ionogram processing capability since September 1968. Since that date, many thousands of topside ionograms have been reduced to electron density profiles, including high altitude ionograms that cannot be processed on an open-loop basis. The FILMCLIP System will continue to be used in the development of new computational software that is described in this report.

ACKNOWLEDGEMENTS

Astrodata expresses its appreciation to Dr. Lawrence Colin and Dr. K. L. Chan of NASA/ARC for their contributions to and guidance of this Study effort.

The Scaling Converter in the FILMCLIP System was designed and built by Ed Haske, Fred Mata, and Elizabeth Starr.

Software development for the FILMCLIP System was provided by Glenn Slike, Jack Lee, William Burns, Wilson Cooper and Geraldine McCulley of Informatics, Inc. Improvements in the N(h) computational software were made by Leonard McCulley of Informatics, Inc.

The contributions of John Kinkel in the development of the Inverse, Matrix, and Horizontal Processing methods are gratefully acknowledged.

Special thanks are due Grace Eckmark for her help in operating the time share computing terminal, and for typing the report. Also thanks to Mary Hillmer for incorporating the corrections and editorial changes. The report was edited by Don Willey.

ABSTRACT

High altitude topside ionospheric sounder data is being provided from both the Alouette II and ISIS-1 satellites. The resulting ionograms are frequently so difficult that open-loop processing techniques are inadequate to cope with the problem. Closed-loop data reduction methods offer a practical solution to the problem of reducing ionograms from high altitude soundings, and at the same time provide improved capability for processing all ionograms.

The FILMCLIP System is a Closed-Loop Ionogram Processor for the reduction of topside ionospheric sounder data recorded on 35mm Film. This system was implemented at NASA/ARC as part of a continuing study to develop methods of automating the data reduction of topside ionograms. The system has worked so well that it is now used nearly full time for film ionogram data reduction on a production basis.

The final data reduction system, which will acquire input data directly from magnetic tape, is known as the TAPECLIP System. Software is being developed for the TAPECLIP System that will use much of the existing software. This software is organized together with new software to exploit the new closed-loop processing techniques that are now available. The TAPECLIP software is being developed and checked out under simulated conditions on the FILMCLIP System.

A number of different methods for computing electron density profiles from topside ionospheric sounder data are presented. The method of Inverse processing is a variation of Jackson's[13] parabolic in log N lamination method. Lamination heights are preselected at equally spaced increments of true height. The scaled X-Trace data is curve fit so that the virtual depth, h'_x, is known for any frequency f on the X-Trace. The plasma frequency f_N is then adjusted in an iterative procedure for each lamination boundary until computed points in the h'(f) plane match the curve fit X-Trace.

Inverse mixed-mode processing provides a way for automatically combining scaled data from both the X and O-Traces to provide a composite electron density N(h) profile.

The Matrix method of ionogram data reduction uses an analytic model of the N(h) profile. In this method the coefficients of a mathematical model of the N(h) profile are adjusted to minimize the error between scaled and computed values of $h'(f_i)$ in a weighted least squares sense.

The method of Horizontal processing provides a means for first order correction of the effects of horizontal gradients of electron density in the satellite orbital plane. The method of Horizontal processing recognizes that the topside sounder ionogram

data and the resultant computed electron density profile are both functions of time due to the satellite velocity with respect to earth based coordinates.

New subroutines that were developed in the course of optimizing some of the computational software as follows:

a. Inverse mapping to specified f

b. Simplified solution of a matrix equation

c. Optimal step size by Golden Section Search

d. Matrix pseudoinverse computation

The weighted least squares approximation method of Mallinckrodt was utilized in the matrix method.

An investigation was conducted to show that values of f_H, ϕ, and h_s between accurately computed points could be obtained by interpolation. The use of interpolation has greatly reduced the computer work load in the determination of values for these parameters.

SECTION I

FILMCLIP SYSTEM

1.0 INTRODUCTION

The FILMCLIP System is a Closed-Loop Ionogram Processor that is now in use on a production basis for the reduction of topside ionospheric sounder data recorded on 35 mm film. The FILMCLIP System was implemented in September 1968 on a minimum cost basis as part of a continuing study for NASA/ARC on advanced equipment and techniques for topside ionospheric sounder data reduction. Since that date many thousands of ionograms have been reduced to electron density profiles, including ionograms from high altitude low density soundings. The latter are almost impossible to reduce on an open loop basis due to the difficulties involved in identifying the portion of the X-Trace due to vertical propagation.

In early 1968, the Computation Division at NASA/ARC made available an IBM 1800 Computer, IBM 2250 Graphic Display Unit, and IBM 360/50 Computer to the Space Sciences Division on a second shift, non-interference basis. With the availability of this computing equipment, Astrodata proposed that a Phase II simulator be implemented at NASA/ARC using the existing Oscar-F data reader and the above computers to provide a closed-loop film ionogram processing capability for investigating advanced processing algorithms, and for reducing film ionograms on a production basis.

The follow-on study for the development of improved topside ionogram data reduction methods has a number of objectives as follows:

a. Improved closed-loop processing methods which take advantage of some of the newer optimization techniques.

b. Relegate more of the ionogram pattern recognition task to computer software.

c. Take better advantage of all the data in an ionogram for production ionogram data reduction.

d. Correction of known deficiencies in existing data reduction methods.

e. Use the FILMCLIP System as a simulator for development of TAPECLIP data reduction software.

1.1 DESCRIPTION

The FILMCLIP System is an on-line interactive computer processing system that has been operational at NASA/ARC since September 1968. A functional block diagram of the FILMCLIP System is shown in Fig. 1-1. The performance characteristics of the FILMCLIP System were defined by Astrodata and the details worked out in a

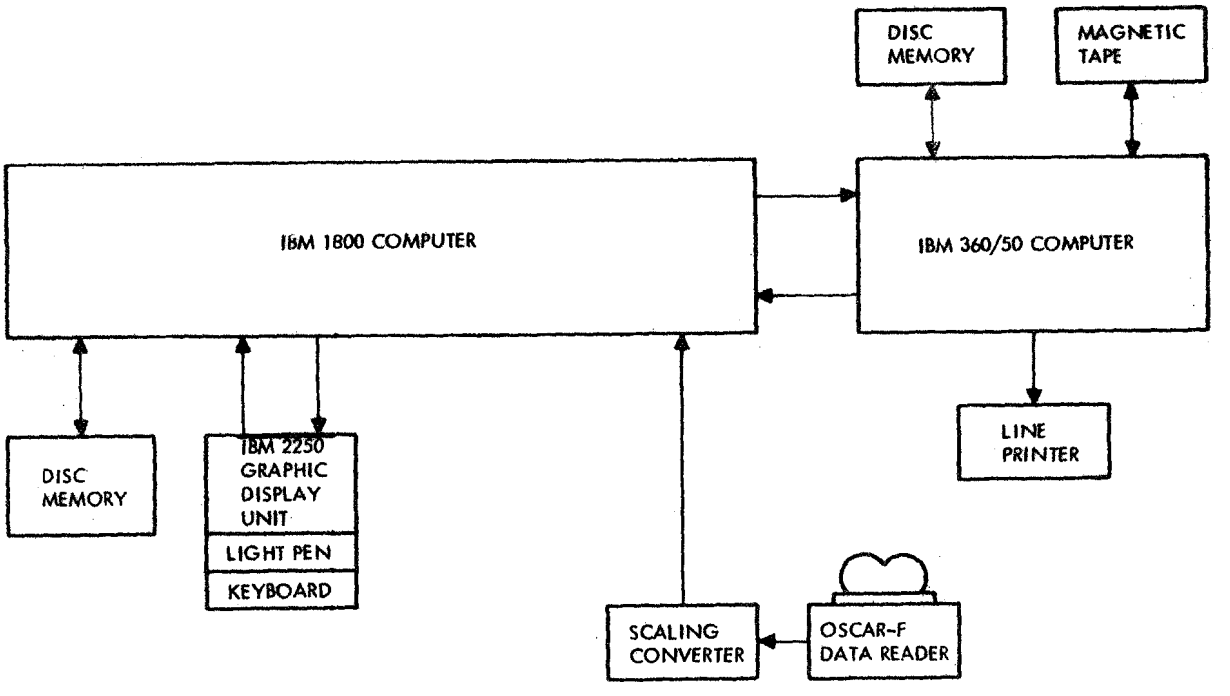


FIGURE 1-1 NASA/ARC FILMCLIP SYSTEM

series of conferences with NASA/ARC and Informatics, Inc. The operational software for the system was provided by Informatics.

A number of technical reports and papers have been written about the FILMCLIP System[1-11] to which the reader is directed for specific detailed information.

The Scaling Converter in the system was built by Astrodata as a piece of interface hardware between the Oscar-F data reader and the IBM 1800 Computer. A functional block diagram of the Scaling Converter is shown in Fig. 1-2. A detailed description of this unit, including logic diagrams and schematics, is contained in the Instruction Manual.[12]

FUNCTIONAL BLOCK DIAGRAM

SCALING CONVERTER

FIGURE 1-2

SECTION II

INTERPOLATION

2.0 INTRODUCTION

Significant savings in computer time can be achieved by simple interpolation of parameter values between accurately computed points, compared with direct computation of the parameters each time new values are needed. The feasibility of interpolating intermediate values of gyrofrequency f_H and satellite height h_s were investigated. The dip angle θ changes so slowly that tests for interpolation accuracy for θ were not included in this investigation.

2.1 INTERPOLATION OF f_H

In the data reduction of topside ionograms to electron density profiles, the X-Trace is used almost exclusively. This is primarily true because the X-Trace is nearly always continuous at the low frequency end down to the cutoff frequency, while the low frequency end of the O-Trace is usually missing. Since the reflection point of the extraordinary wave is dependent on both the electron density N and the flux density B of the earth's magnetic field, it is necessary to compute values of the earth's magnetic field each time the group path integral in eqn. (2-1) [13] is evaluated.

$$h'_x(f) = \int_{h_s}^{h_r} \mu'[N(h), B(h), \theta(h), f]\,dh \qquad (2\text{-}1)$$

where f = sounder pulse frequency

h'_x = virtual depth of reflection of frequency f

h_s = true height of satellite

h_r = true height of reflection of frequency f

μ' = group refractive index

N = electron density in electrons/cm^3

B = earth's magnetic induction in gauss

θ = magnetic dip angle

It is convenient to work with the electron plasma frequency f_N and gyrofrequency f_H so that eqn. (2-1) becomes

$$h'_x(f) = \int_{h_s}^{h_r} \mu'(X, Y, \phi)\,dh \qquad (2\text{-}2)$$

where $X = \left(\frac{f_N}{f}\right)^2$

$Y = \frac{f_H}{f}$

$\phi = 90° - \theta$ for vertical propagation

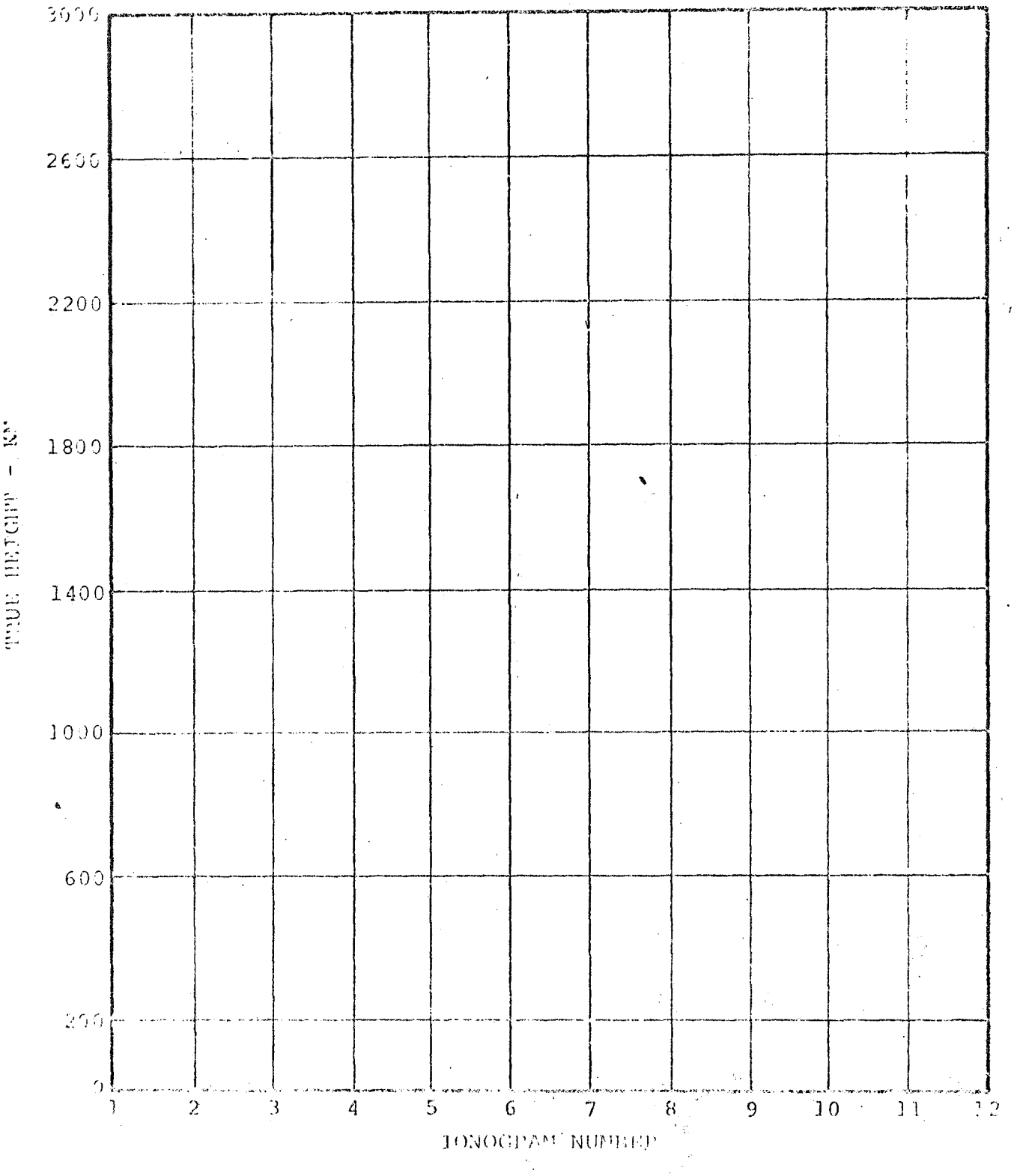
$f_N = \sqrt{N/12400}$

$f_H = 2.8\ B$

Direct computation of the geomagnetic field as a subtask in the evaluation of the group path integral takes a significant amount of computer time. This computation is based on a spherical harmonic expansion representation of the field with coefficients from Daniels and Cain.[17]

In an effort to reduce the number of direct computations of the geomagnetic field, a task was set up to evaluate the possibility of using interpolation as a means for determining the gyro-frequency. The objective of the task was to provide a comparison between directly computed and interpolated values of f_H.

A cross section of the satellite orbital plane was set up as shown in Fig. 2-1 with boundaries 400 km apart vertically between 200 km and 3000 km, and approximately 220 km apart horizontally, which is the average distance the Alouette II satellite travels between corresponding frequency markers from one ionogram to the next. The time between corresponding frequency markers is approximately 31 seconds, and the period of useful data covers roughly one-half the ionogram which is about 1 part in 500 of the orbital period. From this it is reasonable to assume that the velocity of the satellite is essentially constant over the period of the X-Trace, so that linear interpolation of f_H with respect to time at constant height should be feasible.



CROSS SECTION - SATELLITE ORBITAL PLANE

FIGURE 2-1

In the vertical dimension f_H varies approximately as the inverse cube of the distance from the center of the earth. The objective here was to find at what interval it is necessary to compute f_H so that intermediate values could be obtained with adequate accuracy by inverse cube interpolation. The equation for inverse cube interpolation between interval boundaries is

$$f_H = f_{HT}\left(\frac{R + h_T}{R + h}\right)^3 \tag{2-3}$$

where $R = \dfrac{h_T - rh_B}{r - 1}$

$r = \sqrt[3]{f_{HB}/f_{HT}}$

h = height of desired f_H

h_T = height of interval top

h_B = height of interval bottom

$h_B < h < h_T$

f_{HT} = gyrofrequency at h_T

f_{HB} = gyrofrequency at h_B

The detailed results of the f_H interpolation tests are contained in Informatics Technical Notes[3-5]. The worst case error for f_H in these tests was less than 1 part in 3000, and this occurred with inverse cube interpolation at 400 km. From these results we conclude that by computing the gyrofrequency at 400 km intervals vertically, values of f_H can be obtained with adequate accuracy by inverse cube interpolation vertically and linear interpolation horizontally.

Where previously f_H was computed directly for each of 3 or 4 iterations per data point and for 20 to 30 scaled data points per ionogram, the direct computation for f_H can be reduced to a maximum of 8 per ionogram for Alouette II, or a maximum of 10 per ionogram for ISIS-1, which has an apogee of approximately 3600 km.

2.2 INTERPOLATION OF h_s

A separate part of the task, described in paragraph 2.1, was to check the accuracy of linear interpolation of satellite height with respect to time between end points computed once per ionogram.

The position of the satellite as a function of time is computed from an orbital program in which the orbital elements for each satellite are periodically updated. The interpolation tests were run on different portions of the orbit of Alouette II including the case for maximum $\left|\frac{d^2h_s}{dt^2}\right|$.

The detailed results of the h_s interpolation tests are contained in Informatics Technical Notes[3-5]. The worst case error for h_s in these tests was less than 1 part in 2500 in the vicinity of 530 km. From the results of these tests we conclude that it is only necessary to compute the position of the satellite via the orbital program once per ionogram. All other values of h_s can be obtained by linear interpolation with respect to time.

SECTION III

INVERSE PROCESSING

3.0 INTRODUCTION

A method of reducing topside sounder ionograms to electron density profiles using a lamination technique has been developed by Jackson[13]. For convenience this method of data reduction is represented in symbolic form by

$$h'(f) \rightarrow N(h) \tag{3-1}$$

where h' = virtual depth in km due to vertical propagation

f = sounder frequency in MHz

N = electron density in electrons/cm^3

h = true height of reflection in km

Since the N(h) profile can theoretically be computed from the X and O-Traces, and a portion thereof from the Z-Trace, the symbolic representation can be expanded to

$$h'_x(f) \rightarrow N(h) \tag{3-2}$$

$$h'_o(f) \rightarrow N(h) \tag{3-3}$$

$$h'_z(f) \rightarrow N(h) \tag{3-4}$$

From the equation of the group path integral

$$h'(f) = \int_{h_s}^{h_r} \mu'(X,Y,\phi)\,dh \tag{3-5}$$

where μ' = group refractive index

$$X = \left(\frac{f_N}{f}\right)^2$$

$$Y = \frac{f_H}{f}$$

ϕ = angle between earth's magnetic field and direction of vertical propagation

h_r = reflection height

the reverse relationships can be represented by

$$N(h) \rightarrow \overline{h'_x(f)} \tag{3-6}$$

$$N(h) \rightarrow \overline{h'_o(f)} \tag{3-7}$$

$$N(h) \rightarrow \overline{h'_z(f)} \tag{3-8}$$

It follows that an N(h) profile, computed from scaled X-Trace data, can be cross-checked by comparing scaled and computed O-Trace. In symbolic form

$$h'_x(f) \rightarrow N(h) \rightarrow \overline{h'_o(f)} \tag{3-9}$$

This is the method presently used in the FILMCLIP System described in Section I.

For convenience in discussing this method, the following terminology has been adopted:

Forward Processing $\quad h'(f) \rightarrow N(h)$ (3-10)

Reverse Processing $\quad N(h) \rightarrow \overline{h'(f)}$ (3-11)

The method of Inverse Processing, which is discussed in the remainder of this section, uses the Reverse Processing algorithm in an iterative loop.

In the Inverse Processing method, lamination heights are preselected at equally spaced (50 or 100 km) increments of true height. N is assumed to decrease monotonically with h, and the N(h) distribution between laminations is assumed to be parabolic in log N, except the first which is linear in log N. The scaled X-Trace data is curve fit so that the virtual depth h'_x is known for any frequency f on the X-Trace. An iterative procedure is then used to find

$$f_j \ni \left| \frac{h'_x(f_j) - \overline{h'_x(f_j)}}{h'_x(f_j)} \right| < \varepsilon < .001 \qquad (3\text{-}12)$$

When computing an N(h) profile from X-Trace data, the height of reflection is a function of both the electron density N and the gyrofrequency f_H. With f_{Nj} or f_j the trial variable and h_j constant, the only variation in f_{Hj} from one iteration to the next is due to motion of the satellite, and that can be determined by simple linear interpolation with respect to time, as described in Section II. All of the variables on the right hand side of the group integral eqn. (3-5) are known, and therefore $\overline{h'_x(f_j)}$ can be computed directly.

In the Forward Processing method, the gyrofrequency f_{Hj} and true height h_j are both unknown variables on the right hand side of eqn. (3-5). Consequently with f_{Hj} the trial variable, a new value of h_j must be computed in successive iterations until the value of f_{Hj} used in the true height calculation is the same as the actual value of f_{Hj} at altitude h_j.[13]

Lockwood[15] points up another problem with Forward Processing, and that is the iterative solution diverges if the slope of the height of reflection curve has a larger absolute value than the slope of the gyrofrequency curve. As far as we know now, this problem does not exist with the Inverse Processing method.

Enough experimental work has been done at NASA/ARC to demonstrate that the basic Inverse Processing algorithm is valid. Some of the techniques described in this section have not yet been verified.

3.1 DESCRIPTION

In Inverse Processing, lamination heights h_j are selected at equally spaced increments of true height; i.e., 50 or 100 km. X-Trace data, scaled in the conventional manner, is curve fit so that $h'_x(f)$ is defined for any frequency from f_{xs} to the upper frequency limit of the X-Trace on the ionogram being reduced. In the TAPECLIP System, $h'_x(f)$ will be known for every line in the X-Trace so that curve fitting will not be required. A simple linear interpolation will suffice for data points at frequencies that fall between lines.

For each lamination, except the first, the value of N at the lamination bottom at true height h_j may be approximated for the first iteration by extrapolation using a parabolic in log N equation.

The basic parabolic in log N equation[13], with lamination boundaries as shown in Fig. 3-1, is given by

$$h = h_{j-2} + a_{j-1} \log \frac{N}{N_{j-2}} + b_{j-1} \left[\log \frac{N}{N_{j-2}}\right]^2 \qquad (3\text{-}13)$$

$$a_1,\ b_1,\ b_2 = 0$$

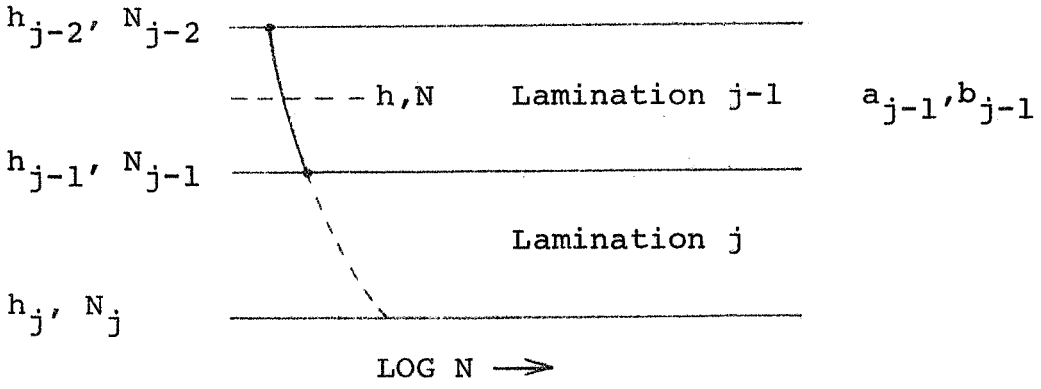


Fig. 3-1. Parabolic In Log N Representation

Rewriting in terms of N with $N = N_j$, the first trial value of electron density for each lamination is given by

$$N_j = N_{j-1}\, e^z \qquad (3\text{-}14)$$

where $$z = \frac{-a_j - \sqrt{a_j^2 - 4b_{j-1}(h_{j-1} - h_j)}}{2b_{j-1}}$$

and $$a_j = a_{j-1} + 2b_{j-1} \log\left(\frac{N_{j-1}}{N_{j-2}}\right)$$

$$j = 3, 4, \ldots, J$$

Since b_j is unknown, we use b_{j-1} as a first approximation. The sounder transmitter frequency f_j is computed from the relation

$$f_j = \frac{f_{Hj} + \sqrt{f_{Hj}^2 + 4f_{Nj}^2}}{2} \tag{3-15}$$

where $f_{Nj} = \sqrt{N_j/12400}$

and the computed value of virtual depth, $\overline{h'_x(f_{j1})}$, is then given by eqn. (3-5).

In the first lamination the variation in electron density is assumed to be linear in log N. The trial value of N at the bottom of the first lamination may be estimated from the equation

$$N_2 = N_1 e^{\left(\frac{h_2 - h_1}{a_2}\right)} \tag{3-16}$$

where a_2 is derived empirically.

An alternative method of establishing initial values of N_j at each lamination boundary was described in Section III of the Study Report, Part I[16]. In this method, values of N_j are linearly extrapolated from the two previous ionograms in the same pass at constant lamination boundary levels. The extrapolation method should work for a large class of ionograms where the gradients at constant true height

$$\left.\frac{\partial N_j}{\partial t}\right|_{h_j=\text{constant}} \tag{3-17}$$

vary slowly throughout the pass.

If $\overline{h'_x(f_{j1})} < h'_x(f_{j1})$, as in Fig. 3-2, the trial value of N_{j1} was too large. For the second iteration choose

$$N_{j2} = N_{j1}(1 \pm \alpha) \tag{3-18}$$

where $\alpha \simeq .01$ with the sign - for $\overline{h'} < h'$
and + for $\overline{h'} > h'$

The second subscript in the above notation is the iteration count. If $\overline{h'_x(f_{j2})} > h'_x(f_{j2})$, the trial value of N_{j2} was too small. A straight line thru points $\overline{h'_x(f_{j1})}$ and $\overline{h'_x(f_{j2})}$ will intersect the X-Trace at f_{j3}, as shown in Fig. 3-2 where the first two points are on opposite sides of the X-Trace, or in Fig. 3-3 where the first two points fall on the same side of the X-Trace.

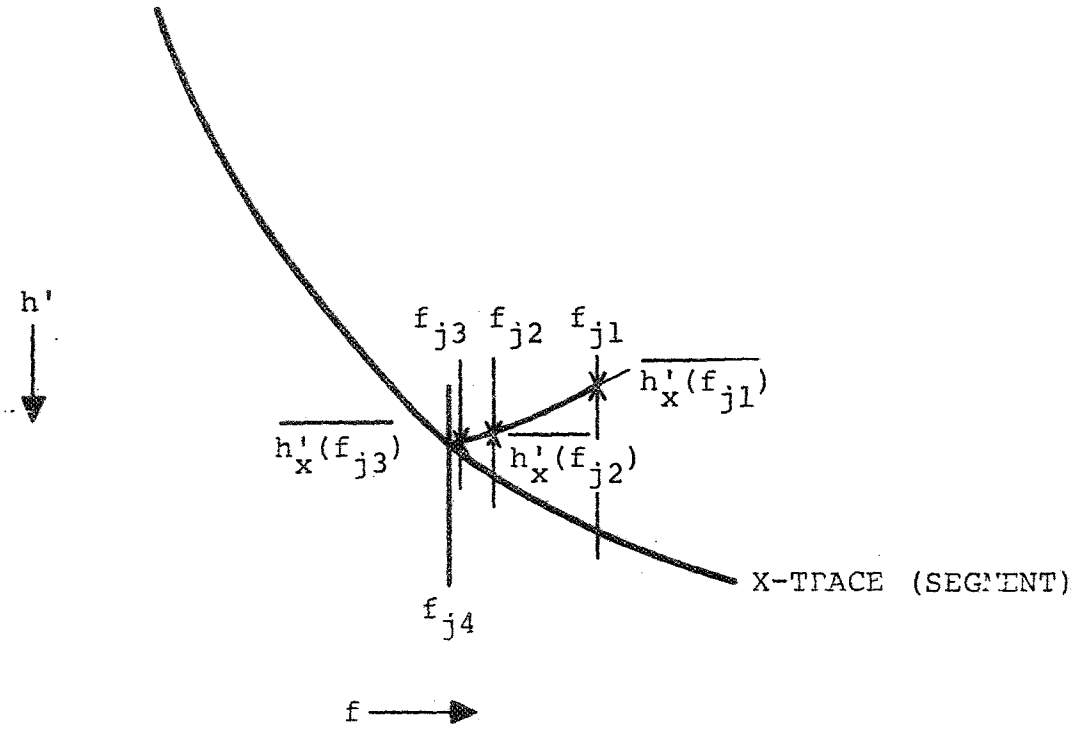

h'
f_{j3}
f_{j2}
f_{j1}
$\overline{h'_x(f_{j1})}$
$\overline{h'_x(f_{j3})}$
$\overline{h'_x(f_{j2})}$
X-TRACE (SEGMENT)
f_{j4}
f

FIG. 3-3

For the third iteration,

$$f_{Nj} = \sqrt{f_j(f_j - f_{Hj})} \tag{3-19}$$

and from eqn. (3-5) a third point $\overline{h'_x(f_{j3})}$ is computed. A segment of a circle passing thru the three computed values of $\overline{h'_x(f_j)}$ will intersect the X-Trace at frequency f_{j4} from which it should be possible to compute the final value of Nj from eqn. (3-19) and the relation

$$N(h_j) = 12400\ f_{Nj}^{\ 2} \tag{3-20}$$

without computing $\overline{h'_x(f_j)}$ a fourth time from eqn. (3-5). Of course this last conclusion needs to be verified, but if it is valid it will help considerably in reducing the computation time required to compute an N(h) profile.

If the criterion of eqn. (3-12) is met on any of the first three iterations, the iterative sequence for that lamination is terminated and the program continues to the next lamination.

The lamination bottom of the last lamination in the N(h) profile will be dependent on the scaled data point $h'_x(f)$ at the highest frequency on the X-Trace. For this case the height of reflection is unknown, and so the Forward Processing algorithm is required.

The objectives of the Inverse Processing method are as follows:

a. Minimize the computer time required to compute an N(h) profile.

b. Simplify the implementation of mixed-mode processing.

c. Optimize the spacing between lamination boundaries of the N(h) profile.

d. Make the lamination boundaries the same for all ionograms for convenience in interpolating and extrapolating values of N(h), f_H and ϕ.

3.2 STEP SEQUENCE

The following is a description of the sequence of steps for computing an N(h) profile from scaled X-Trace data by the Inverse Processing method. The following step numbers relate to the processing blocks in the flow diagram of Fig. 3-4.

1. Scale the X-Trace in conventional manner.

2. Curve fit the X-Trace with a suitable function.

3. Compute the plasma frequency at satellite height

$$f_N(h_1) = \sqrt{f_1(f_1 - f_{H1})}$$

4. Select lamination boundaries h_j at equally spaced increments of true height.

   a. 50 km $\leq (h_1 - h_2) <$ 100 km for step 4b.

   $<$ 150 km for step 4c.

   b. IF (h_1 .LE. 1500.) h_j = m(50)km

   c. IF (h_1 .GT. 1500.) h_j = m(100)km

   $j = 2, 3, \ldots, J$, m an integer

5. Initialize the iteration count, ITCNT = 0

6. Select trial value for $f_{Nj} = \sqrt{N_j/12400}$

6.1 First iteration

a. First lamination $N_2 = N_1 e^{\left(\frac{h_2 - h_1}{a_2}\right)}$

where a_2 is derived empirically

b. Subsequent laminations $N_j = N_{j-1} e^{z}$

where $z = \dfrac{-a_j - \sqrt{a_j^2 - 4b_{j-1}(h_{j-1} - h_j)}}{2b_{j-1}}$

$$a_j = a_{j-1} + 2b_{j-1} \log\left(\frac{N_{j-1}}{N_{j-2}}\right)$$

$$a_1,\ b_1,\ b_2 = 0$$

$$j = 3,\ 4,\ \ldots.,\ J$$

6.2 Second iteration $N_{j2} = N_{j1}(1 \pm \alpha)$

where $\alpha \simeq .01$ with sign - for $\overline{h'} < h'$

\+ for $\overline{h'} > h'$

6.3 Compute $\overline{h'(f_{j1})}$ and $\overline{h'(f_{j2})}$, processing Fig. 3-4 blocks 7 thru 19.

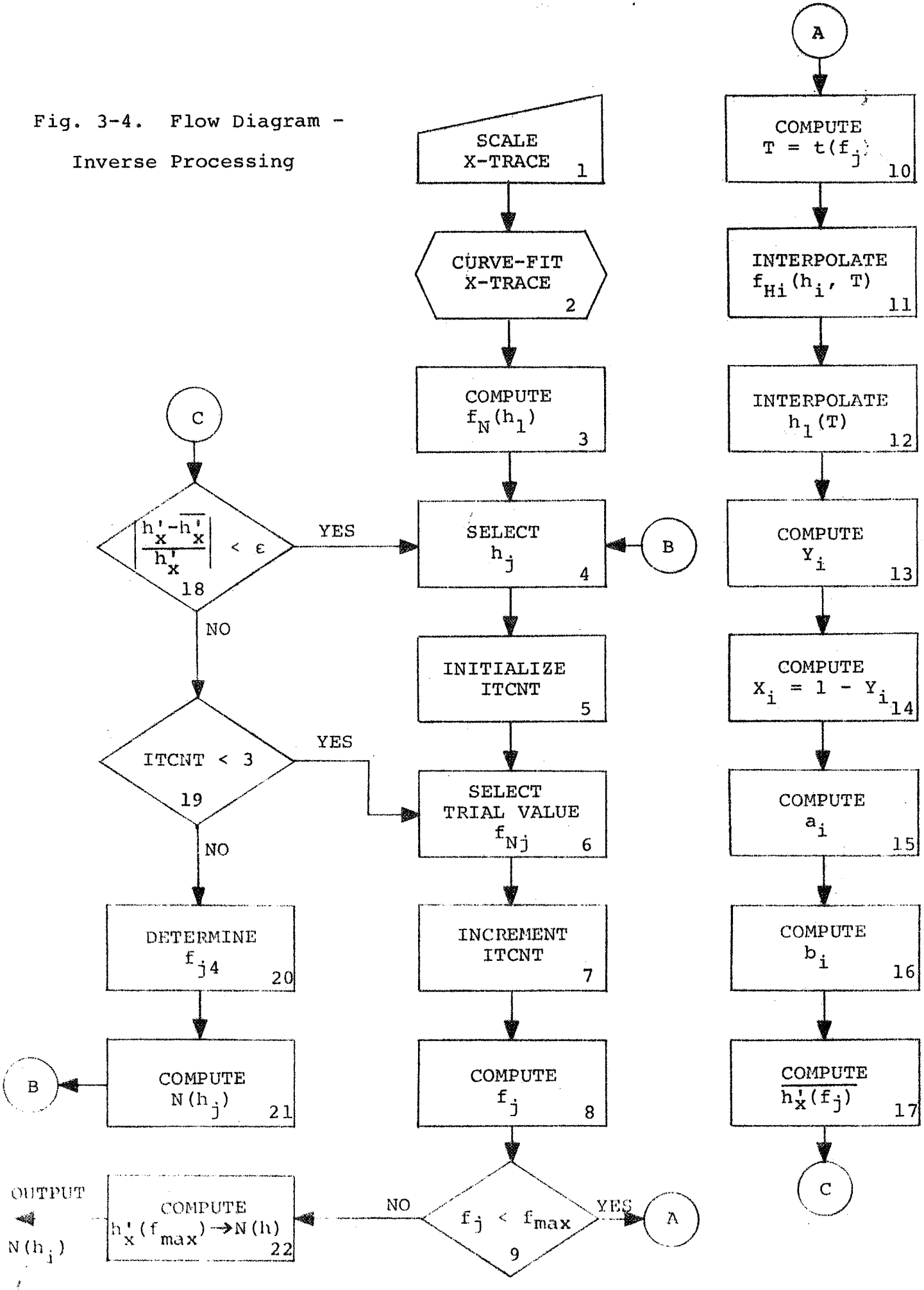
6.4 Compute equation of a line thru the two points in step 6.3.

$$s = \frac{\overline{h'_{j2}} - \overline{h'_{j1}}}{f_{j2} - f_{j1}} \qquad \text{slope}$$

$$r = \overline{h_{j1}} - sf_{j1} \qquad \text{intercept}$$

$$\overline{h'_x} = sf + r$$

Fig. 3-4. Flow Diagram - Inverse Processing



Solve for frequency f_{j3} at the intersection of the straight line and the X-Trace.

6.5 Compute $f_{Nj} = \sqrt{f_{j3}(f_{j3} - f_{Hj})}$

6.6 Compute $\overline{h'_x(f_{j3})}$, processing Fig. 3-4 blocks 7 thru 19.

7. Increment the iteration counter ITCNT = ITCNT + 1

8. Compute $f_j = \dfrac{f_{Hj} + \sqrt{f_{Hj}^2 + 4f_{Nj}^2}}{2}$

9. Check if $f_j < f_{max}$

   a. YES, the trial frequency is still on the X-Trace. Compute $\overline{h'_x(f_j)}$ on branch to Fig. 3-4, block 10.

   b. NO, there is no more X-Trace data. Branch to block 22, Fig. 3-4.

10. Compute $T_j = t(f_j)$ from time and frequency marker calibration data.

11. At constant h_i, f_{Hi} with i = 1, 2, ..., j is computed by linear interpolation with respect to time, as described in Section II, to compensate for the change in satellite position.

12. In the first lamination the slope of the N(h) curve, which is linear in log N, is given by:

$$a_2 = \frac{h_2 - h_1}{\log \frac{N_2}{N_1}} \qquad \text{Eqn. (2),}^{[18]}$$

where h_1 = satellite height

As the computation of $N(h_j)$ progresses for each lamination boundary, the change in satellite height can be determined by linear interpolation with respect to time, given the slope of the orbit for that ionogram, $\frac{\Delta h_s}{\Delta t}$. The new value of $N(h_{1j}, T_j)$ for the j^{th} lamination is computed from

$$N_{1j} = N_2 e^{\left(\frac{h_{1j}-h_2}{a_2}\right)}$$

13. Compute $Y_i = \frac{f_{Hi}}{f_j}$

14. Compute $X_i = 1 - Y_i$

$i = 1, 2, \ldots, j$

15. $a_i = a_{i-1} + 2b_{i-1} \log\left(\frac{N_{i-1}}{N_{i-2}}\right)$

16. $b_i = \frac{\left(h_i - h_{i-1}\right) - a_i \log\left(\frac{N_i}{N_{i-1}}\right)}{\log\left(\frac{N_i}{N_{i-1}}\right)}$

$i = 3, 4, \ldots, j$

$a_1, b_1, b_2 = 0$

Eqn. (3),(4),[18]

17. Compute $\overline{h'_x(f_j)}$ using eqn. (7),[18].

18. Compute error ratio and compare with limiting value

$$\left| \frac{h'_x(f_j) - \overline{h'_x(f_j)}}{h'_x(f_j)} \right| < .001$$

19. Compare iteration count with integer 3.

    a. If <, continue next iteration, step 6.

    b. If $\geq$, compute $N(h_j)$, steps 20 and 21.

20. Pass a smooth curve (segment of a circle) thru points $\overline{h'_x(f_{jk})}$, k = 1, 2, 3, intersecting the X-Trace at f_{j4}.

21. Compute $f_{Nj} = \sqrt{f_{j4}(f_{j4} - f_{Hj})}$

$$N(h_j) = 12400\ f_{Nj}^2$$

22. Compute $N(h_J)$ by Forward Processing, where h_J is the bottom of the last lamination. Output $N(h_i)$,
    i = 1, 2, ..., j
    j = J

3.3 MIXED MODE

The useful portions of the O and Z-Traces can be used in the Inverse Processing method as well as X-Trace data as described in in paragraph 3.1.

X, O, and Z-Trace data are scaled in the conventional manner and curve fit so that $h'_x(f)$, $h'_o(f)$, and $h'_z(f)$ are defined for all frequencies on the useful portions of their respective traces. Mixed-Mode Inverse Processing begins with X-Trace data as in paragraph 3.1. At each lamination boundary at the point in the program where the sounder transmitter frequency f_j is computed, as in eqn. (3-15), the corresponding frequencies for the O and Z-Traces are also computed.

$$\overline{f}_{xj} = \frac{f_{Hj} + \sqrt{f_{Hj}^2 + 4f_{Nj}^2}}{2} \tag{3-21}$$

$$\overline{f}_{oj} = f_{Nj} \tag{3-22}$$

$$\overline{f}_{zj} = \overline{f}_{xj} - f_{Hj} \tag{3-23}$$

A comparison is made with the curve fit O and Z-Trace segments to see if there is an $h'_o(f)$ at frequency $\overline{f}_{oj}$ or an $h'_z(f)$ at frequency $\overline{f}_{zj}$. For this example, assume that an $h'_o(\overline{f}_{oj})$ does exist. The Reverse computation

$$N(h_j) \longrightarrow \overline{h'_o(f_{j1})} \tag{3-24}$$

gives the first point for lamination j on the h'(f) plane. The iterative sequence continues as in paragraph 3.1, except for the third iteration where

$$f_{Nj} = \overline{f}_j \tag{3-25}$$

from which a third point $\overline{h'_o(f_{j3})}$ is computed. A segment of a circle passing thru the three computed values of $\overline{h'_o(f_j)}$ will intersect the O-Trace at frequency f_{j4}, from which it should be possible to compute the final value of N_j from eqn. (3-25) and eqn. (3-20).

To compute $N(h_j)$ with respect to curve fit Z-Trace data use eqn. (3-23) and the Reverse computation

$$N(h_j) \longrightarrow \overline{h'_z(f_j)} \tag{3-26}$$

At any lamination boundary h_j, it would be possible to compute an $N(h_j)$ with respect to each of the X, O, or Z-Traces, assuming the traces have been scaled at the related sounder pulse frequencies. The value of $N(h_j)$ would be slightly different for each trace and would depend of course on the delay contribution of the previous laminations.

On the basis that the O-Trace is usually thinner than either the X or Z-Trace, the program logic for Mixed Mode Inverse Processing might be to compute $N(h_j)$ with respect to the O-Trace whenever possible, otherwise use X-Trace data. When processing with

respect to the O-Trace, corresponding $\overline{h'_x(f_j)}$ points could be computed to assist in defining the vertical reflection portion of a difficult X-Trace.

SECTION IV

MATRIX PROCESSING

4.0 INTRODUCTION

One of the problems an operator faces when processing ionograms with the FILMCLIP System occurs after one processing iteration represented by

$$h'_x(f_j) \rightarrow N(h_j) \rightarrow \overline{h'_o(f_j)}$$

If the computed O-Trace data points $\overline{h'_o(f_j)}$ don't match closely enough the identifiable O-Trace on the ionogram, the operator must change some of the scaled data points on the X-Trace in an effort to modify the N(h) profile so that the computed $\overline{h'_o(f_j)}$ data points will match the O-Trace more closely. This is really a very complex, multi-dimensional, non-linear problem that a human being is not equipped to handle, except by trial and error. With the FILMCLIP System an operator can make adjustments in the positions of the scaled data points on the X-Trace and observe the results referred to the O-Trace in successive iterations until a suitable match is achieved. With experience, operators become quite skilled in making adjustments in this iterative loop, but it still remains a time consuming and inefficient procedure.

The matrix method provides a logical means whereby the computer can use all the normally available information in an ionogram to compute an N(h) profile in a weighted least squares sense.

The computation time to compute an N(h) profile by the matrix method will increase substantially over the time required for either the Forward or Inverse methods. However, it will take the operator out of the iterative loop for closed-loop processing so that there may be an overall reduction in elapsed time. In addition, the matrix method offers some unique advantages not available in other methods.

The matrix method of ionogram data reduction uses an analytic model of the N(h) profile rather than the lamination model of Jackson[13]. The concept behind matrix processing is to set up a mathematical model of the N(h) profile and then adjust the coefficients of the mathematical model to minimize the error between scaled and computed values of $h'(f_i)$ in a weighted least squares sense.

A set of scaled $h'(f_i)$ data points from an ionogram is the input prescription vector to the matrix program. The data points can be scaled from both the X and O-Traces and even the Z-Trace when it is available. A weighting coefficient is assigned to each data point using an arbitrary scale from 1 to 10. Good data

points are weighted higher than questionable data points so that the resultant N(h) profile is influenced more by the data points in which the operator has greater confidence and to a lesser extent by data points with a lower weighting factor.

The particular model used for matrix processing is a ratio of polynomials of the type used in synthesizing electrical filter transfer functions. It is perhaps one of those fortuitous circumstances, but when a set of N(h) values were input to a slightly modified Astrodata proprietary program for filter synthesis, a curve fit within 1% was achieved the first time the right number of poles and zeros were specified.

With the matrix method it is desirable to minimize the number of coefficients required to model the N(h) profile, because some matrix operations on the computer go up as m^3 where m is the order of a square matrix. The roots of the numerator and denominator can be plotted as poles and zeros in the complex s-plane. A plot of the poles and zeros for one model is shown in Fig. 4-1. In this model there are four conjugate complex poles, two real poles, and one zero at the origin. Other models investigated have used five or six complex poles with from two to five zeros at the origin and no real poles. The number of adjustable coefficients required for these models varies from 10 to 12. Kinkel[21] proposed a model with 5 variables which turned out to be an insufficient number.

The choice of the particular zero/pole representation for an N(h) profile depends on a number of factors, such as the ratio of minimum to maximum electron density and the shape of the curve. A little experience in curve-fitting a variety of N(h) profiles, using Program NHMODEL in Section 4.4, will provide the basis for selecting the initial values for coefficients of the model used in the matrix method.

4.1 DESCRIPTION

The general form of the analytic model used in matrix processing is shown in eqn. (4-1).

$$N(s) = \frac{\rho s^n}{(s+\sigma_\ell)\left[(s+\sigma_m)^2 + \beta_m^2\right]} \qquad (4\text{-}1)$$

where $s = j(\eta - h)$

h = true height above mean sea level in km $\times 10^3$

η = a constant > maximum satellite height in same units as h

ρ = scale factor

σ = real component of pole position

β = imaginary component of pole position

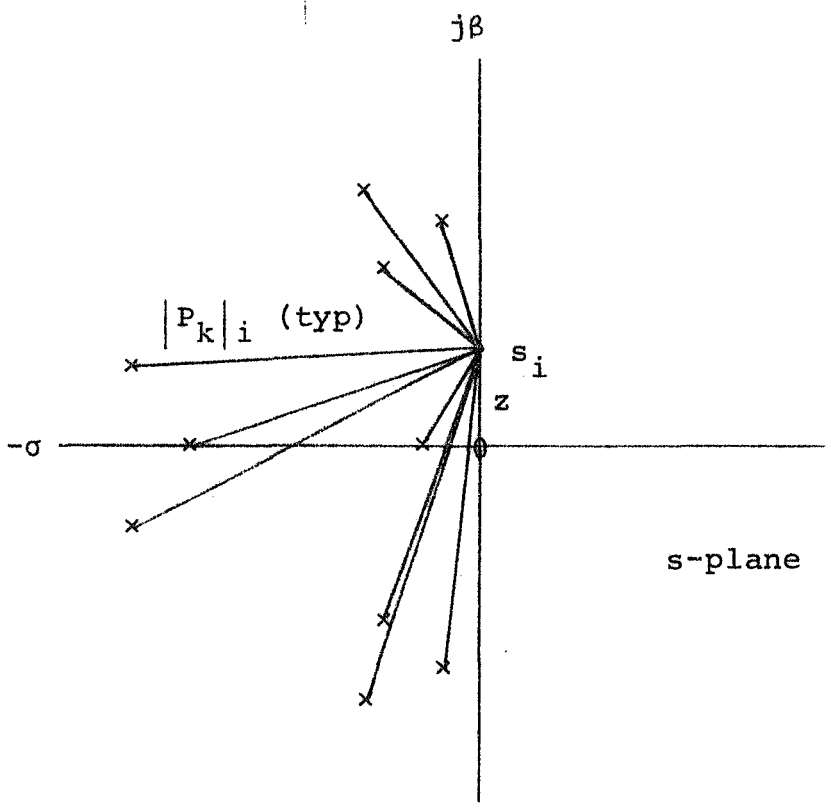
$j = \sqrt{-1}$

$\ell = 0, 1, 2$ index of real poles

$m = 1, 2, \ldots, M$ index of complex poles

$n = 1, 2, \ldots, 5$ index of zeros at origin

Eqn. (4-1) is a representation of an N(h) profile in the complex s-plane using two real poles, four conjugate complex poles, and a zero at the origin. A representative distribution of poles and the zero are shown in Fig. 4-1. In this representation, only the magnitude of the function is considered. The denominator of the function for a given value of s_i on the positive imaginary



ZERO/POLE REPRESENTATION OF N(h) PROFILE

FIG. 4-1

axis, is computed as the product set of all the vector magnitudes, $|p_k|_i$. The numerator of the function is the vector magnitude z along the imaginary axis from s_i to the origin multiplied by the scale factor ρ.

Then the value of points $N(s_i)$ is given by eqn. (4-2).

$$N(s_i) = \frac{\rho \, z_i^{\,n}}{\prod_{k=1}^{K} |p_k|_i} \tag{4-2}$$

where $s_i = j(\eta - h_i)$ as in eqn. (4-1)

$i = 1, 2, \ldots, I$

Define a column vector

$$\underline{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_K)^T \tag{4-3}$$

$$\begin{aligned}
\alpha_1 &= \sigma_1 \\
\alpha_2 &= \sigma_2 \\
\alpha_3 &= \sigma_3 \\
\alpha_4 &= \beta_3 \\
&\vdots \\
\alpha_{K-1} &= \sigma_M \\
\alpha_K &= \beta_M
\end{aligned}$$

All of the variables α and β in eqn. (4-1) are represented by $\underline{\alpha}$.

In this notation

I = number of scaled data points in the prescription vector

K = number of degrees of freedom in the mathematical model of the N(h) profile

An initial set of values of the $\underline{\alpha}$ vector is established to represent a trial function of $N(h_i, \underline{\alpha})$ for a set of scaled $h_i'(f_i)$ data points from an ionogram, where i = 1, 2, ..., I. The $h_i'(f_i)$ will be a composite set of points scaled from both the X-Trace and O-Trace, and even the Z-Trace if it is available.

We wish to approximate the set of scaled data points by a vector $H(\overline{f}_i, \underline{\alpha})$ in the h'(f) plane which is a mapping of points from the approximation $N(h_i, \underline{\alpha})$ in the N(h) plane as shown in Fig. 4-2.[23] These points are computed using Reverse Processing as defined in Section III. The value of h_i for each point $N(h_i, \underline{\alpha})$ is computed by successive approximation in an iterative loop so that

$$\left| \frac{f_i - \overline{f}_i}{f_i} \right| < \epsilon < .001 \qquad (4\text{-}4)$$

This iterative loop is described in paragraph 4.2.

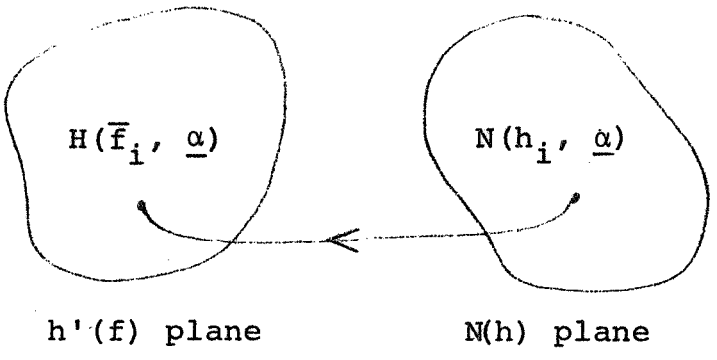


Fig. 4-2. Mapping of Corresponding Points in the h'(f) and N(h) Planes

The differential of the function $H(\bar{f}_i, \underline{\alpha})$ can be approximated by a small increment on each α_k as in eqn. (4-5).

$$\Delta H(\bar{f}_i, \underline{\alpha}) = \sum_{k=1}^{K} \frac{\partial H(\bar{f}_i, \underline{\alpha})}{\partial \alpha_k} \Delta\alpha_k \qquad (4\text{-}5)$$

$$i = 1, 2, \ldots, I$$

Eqn. (4-3) expressed in gradient notation becomes

$$\Delta H(\bar{f}_i, \underline{\alpha}) = \nabla_{\underline{\alpha}} H(\bar{f}_i, \underline{\alpha}) \Delta\underline{\alpha} \qquad (4\text{-}6)$$

$$i = 1, 2, \ldots, I$$

where $$\nabla_{\underline{\alpha}} H(\bar{f}_i, \underline{\alpha}) = \frac{\partial H(\bar{f}_i, \underline{\alpha})}{\partial \alpha_1}, \ldots, \frac{\partial H(\bar{f}_i, \underline{\alpha})}{\partial \alpha_K}$$

$$\Delta\underline{\alpha} = (\Delta\alpha_1, \ldots, \Delta\alpha_K)^T$$

We wish to find

$$\Delta H(\bar{f}_i, \underline{\alpha}) \ni h'(f_i) - H(\bar{f}_i, \underline{\alpha}) - \Delta H(\bar{f}_i, \underline{\alpha}) = 0 \qquad (4\text{-}7)$$

$$i = 1, 2, \ldots, I$$

Rearranging terms and normalizing by dividing both sides by $H(\bar{f}_i, \underline{\alpha})$

$$\frac{h'(f_i) - H(\bar{f}_i, \underline{\alpha})}{H(\bar{f}_i, \underline{\alpha})} = \frac{\Delta H(\bar{f}_i, \underline{\alpha})}{H(\bar{f}_i, \underline{\alpha})} \qquad (4\text{-}8)$$

Combining eqn. (4-6) and eqn. (4-8) and expanding in matrix form gives

$$\begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_I \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1K} \\ a_{21} & a_{22} & \cdots & a_{2K} \\ & & \vdots & \\ a_{I1} & a_{I2} & \cdots & a_{IK} \end{bmatrix} \begin{bmatrix} \Delta\alpha_1 \\ \Delta\alpha_2 \\ \vdots \\ \Delta\alpha_K \end{bmatrix} \qquad (4\text{-}9)$$

where $$e_i = \frac{h'(f_i) - H(\bar{f}_i, \underline{\alpha})}{H(\bar{f}_i, \underline{\alpha})}$$

$$a_{ik} = \frac{\partial H(\bar{f}_i, \underline{\alpha})}{H(\bar{f}_i, \underline{\alpha})\, \partial\alpha_k}$$

$$i = 2, 3, \ldots, I$$

$$k = 1, 2, \ldots, K$$

For i = 1,

$$h'(f_1) = 0$$

where $f_1 = f_{xs}$

consequently e_1 and a_{1k}, k = 1, 2, ..., K, must be redefined in terms of frequency because ΔH has no meaning at satellite height. For this case

$$e_1 = \frac{f_x(h_1) - F(h_1, \underline{\alpha})}{F(h_1, \underline{\alpha})}$$

$$a_{1k} = \frac{\partial F(h_1, \underline{\alpha})}{F(h_1, \underline{\alpha})\,\partial \alpha_k}$$

In compact matrix notation

$$\underline{E} = \underline{A}\ \Delta\underline{\alpha} \tag{4-10}$$

In eqn. (4-9) there are I equations with K unknowns. With I > K there are more equations than unknowns since typically

$$20 \leq I \leq 30$$

and $$10 \leq K \leq 12$$

A least squares fit solution[19] can be obtained by first multiplying both sides of eqn. (4-10) by $\underline{A}^T$, which gives

$$\underline{A}^T\underline{E} = [\underline{A}^T\underline{A}]\Delta\underline{\alpha} \tag{4-11}$$

where T = the transpose

Then a least squares estimate of the adjustment vector $\Delta\underline{\alpha}$ is given by

$$\Delta\hat{\underline{\alpha}} = [\underline{A}^T\underline{A}]^{-1}\ \underline{A}^T\underline{E} \tag{4-12}$$

We wish to assign a weighting coefficient, w_{ii}, to each scaled $h'(f_i)$ data point in order that good data points exert a greater influence on the least squares approximation of $N(h, \underline{\alpha})$ than questionable data points. Define a diagonal matrix of weights as follows:

$$\underline{W} = \begin{bmatrix} w_{11} & & & & \\ & w_{22} & & 0 & \\ & & w_{33} & & \\ & & & \ddots & \\ & 0 & & & \ddots \\ & & & & & w_{II} \end{bmatrix} \tag{4-13}$$

The weighting matrix $\underline{W}$ is incorporated in the least squares fit solution of $\Delta\hat{\underline{\alpha}}$ to give a weighted least squares adjustment vector as follows:

$$\Delta\hat{\underline{\alpha}} = \left[\underline{A}^T\underline{WA}\right]^{-1}\ \underline{A}^T\underline{WE} \tag{4-14}$$

For each iteration then

$$\underline{\alpha}^{n+1} = \underline{\alpha}^{n} + \Delta\hat{\underline{\alpha}} \tag{4-15}$$

where n = iteration count

which gives a new set of coefficients for the analytic model $N(h, \underline{\alpha})$. The iterative cycle continues until

$$\left|\frac{\Delta\alpha_k}{\alpha_k}\right|_{max} < \varepsilon < .001 \tag{4-16}$$

A more direct measure of how well the computed data points $H(\bar{f}_i, \underline{\alpha})$ match the scaled data points $h'(f_i)$ might be obtained by defining a cost functional J as the sum set of the weighted, normalized error squared.

$$J = \sum_{i=1}^{I} d_i \tag{4-17}$$

where $\underline{D} = [d_1, d_2, \ldots, d_I]^T = \underline{E}^T\underline{W}\underline{E}$

The iterative cycle would then continue until

$$\left|\frac{J^{n+1} - J^n}{J^n}\right| < \varepsilon \tag{4-18}$$

where n = iteration count

A flow diagram showing the major processing steps in a program for the matrix method is shown in Fig. 4-3.

4.2 STEP SEQUENCE

The following is a description of the sequence of steps for computing an N(h) profile from scaled X-Trace data by the Matrix Processing method. The following step numbers relate to the processing blocks in the flow diagram of Fig. 4-3.

1. Scale the X, O, and Z-Traces of an ionogram in conventional manner.

2. Assign a weighting coefficient to each scaled data point.

3. Establish initial values of coefficients for $\underline{\alpha}$. Program NHMODEL has been developed for doing this and is described in paragraph 4.4.

4. Compute values of h_i corresponding to frequencies f_i of scaled data points $h'(f_i)$ such that

$$\left| \frac{f_i - \overline{f_i}}{f_i} \right| < \varepsilon$$

This is accomplished by a one-dimensional search[23] using successive approximations and is described in paragraph 4.3. This makes possible comparison of computed and scaled data points at the same frequency in the h'(f)-plane.

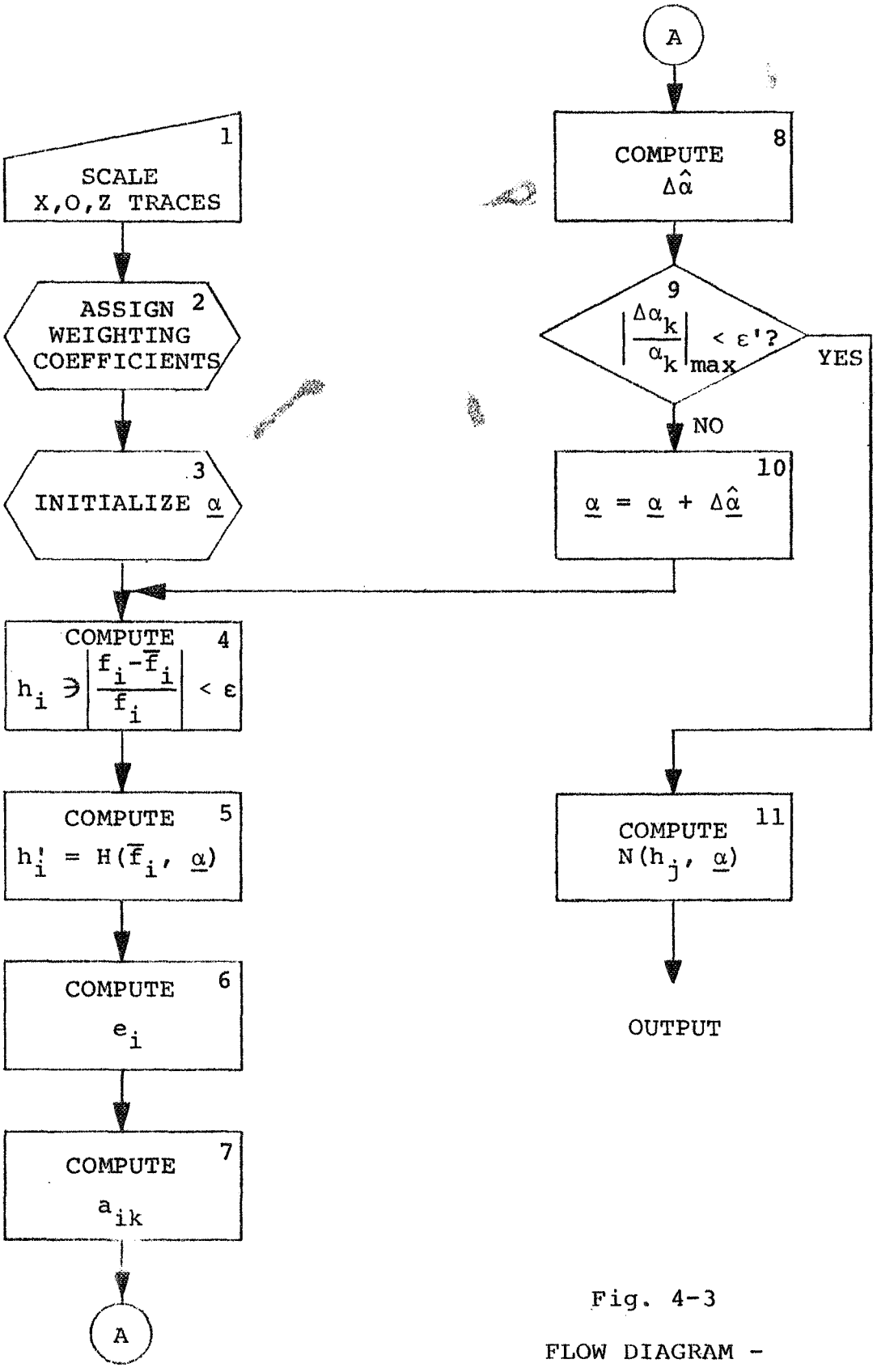


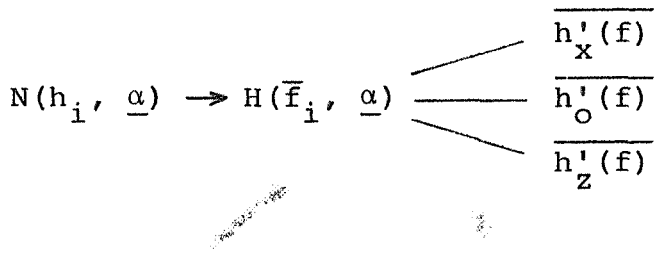
Fig. 4-3

FLOW DIAGRAM -

N(h) Processing

Matrix Method

5. Compute data points $H(\bar{f}_i, \underline{\alpha})$ in the h'(f)-plane using the Reverse Processing algorithm of Section III.



The electron density profile $N(h_i, \underline{\alpha})$ for this computation is obtained from the mathematical model of eqn. (4-1). The computation of $N(h_i, \underline{\alpha})$ is handled by SUBROUTINE CALC and is described in paragraph 4.4.

6. Compute the elements e_i of discrepancy vector $\underline{E}$ as defined in eqn. (4-7).

7. Compute the elements of matrix $\underline{A}$. This is the matrix of partial derivatives of $H(\bar{f}_i, \underline{\alpha})$ with respect to $\underline{\alpha}$, with each element a_{ik} as defined in eqn. (4-7). The term $\partial\alpha_k$ is approximated by perturbing the elements of $\underline{\alpha}$ by 0.1%, one at a time.

8. Compute the elements $\Delta\alpha_k$ of the adjustment vector $\Delta\hat{\underline{\alpha}}$, which is the solution of matrix eqn. (4-12). This is handled by SUBROUTINE MXV which is described in paragraph 4.4.

9. This step is a decision block to determine whether or not the latest adjustment vector will make a significant change in the coefficients of the mathematical model of the N(h) profile. If the maximum absolute value of all the change ratios of vector $\underline{\alpha}$ are less than ε, written as

$$\left|\frac{\Delta\alpha_k}{\alpha_k}\right|_{max} < \varepsilon, \qquad k = 1, 2, \ldots, K$$

   a. NO, branch to Step 10. Repeat steps 4 thru 9.

   b. YES, a weighted least squares solution for $N(h_i, \underline{\alpha})$ has been obtained.

10. Add the adjustment vector $\Delta\hat{\underline{\alpha}}$ to the state variables $\underline{\alpha}$ and repeat steps 4 thru 9.

11. Output $N(h_j, \underline{\alpha})$ at equally spaced increments of true height h_j.

4.3 COMPUTATION OF h_i

The matrix method of computing an N(h) profile requires that comparisons of h'(f) and $H(\bar{f}_i, \underline{\alpha})$ be made at the same frequency. Since only portions of an O-Trace are usually available, and we would like to avoid the extra step of curve fitting a scaled X-Trace, a method is described in this section whereby values of true height h_i are computed such that computed data points $H(\bar{f}_i, \underline{\alpha})$ will occur at the same frequencies as the corresponding scaled data points $h'(f_i)$.[23]

The frequency, $\bar{f}_i$, of a computed data point $H(\bar{f}_i, \underline{\alpha})$ for the X, O, and Z-Traces is a function of plasma frequency f_N, and for the X and Z-Traces is also dependent on the gyro frequency f_H, as follows[13]:

$$f_x = \frac{f_H + \sqrt{f_H^2 + 4f_N^2}}{2} \tag{4-19}$$

$$f_o = f_N \tag{4-20}$$

$$f_z = f_x - f_H \tag{4-21}$$

$$f_N = \sqrt{N/12400} \tag{4-22}$$

$$f_H = 2.8B \tag{4-23}$$

where N = ionospheric electron density (electrons/cm^3)

B = induction (gauss) earth magnetic field

f = frequency in MHz

Since N and B are monotone decreasing in h, f_N and f_H are also monotone decreasing, and therefore the computed frequency $\bar{f}_i$, computed from eqn. (4-19) through eqn. (4-21), is monotone decreasing in h. This property makes it possible to use a technique of successive approximation for finding h_i such that

$$\left| \frac{f_i - \bar{f}_i}{f_i} \right| < \varepsilon < .001 \tag{4-24}$$

For the general case

$$h_i \leq h_s = \text{satellite height} \tag{4-25}$$

so that the first trial value of h_i will be

$$h_{i1} = \frac{h_s}{2} \tag{4-26}$$

If $\bar{f}_{i1} > f_i$, then $h_{i2} > h_{i1}$. On the first iteration, in this example, h_i is in the upper half of the interval h_s to h = 0. On successive iterations h_i is bounded by 1/4, 1/8, etc., of the total height interval until the conditions of eqn. (4-22) are met. At each trial value of h_i, f_N is computed from eqn. (4-2) and eqn. (4-22).

$$N(\eta - h_i) = \rho \frac{z_i^n}{\prod_{k=1}^{K} |p_k|_i} \tag{4-2}$$

where the variables are the same as previously defined. The value of f_H at h_i is obtained by inverse cube interpolation (refer to Section II) at the satellite position, which is known from $t(f_i)$ and the topside sounder orbital parameters.

The following is a brief description of the various processing steps in sequence as they relate to computing h_i prior to obtaining $H(\bar{f}_i, \underline{\alpha})$ by Reverse Processing. The following step numbers relate to the processing blocks in the flow diagram of Fig. 4-4.

1. Input f_i, $t(f_1)$, h_s, and trace designator (X, O, or Z-Trace)

2. Compute trial value of h_i

3. Compute f_N, f_H, $\bar{f}_i$, Δh. The value of $f_H(h)$ is obtained by inverse cube interpolation vertically with respect to true height and linear interpolation horizontally with respect to time.

4. If $\left|\frac{f_i - \bar{f}_i}{f_i}\right| < \varepsilon$

   NO, go to step 5.

   YES, $h_i = h$. RETURN to calling program.

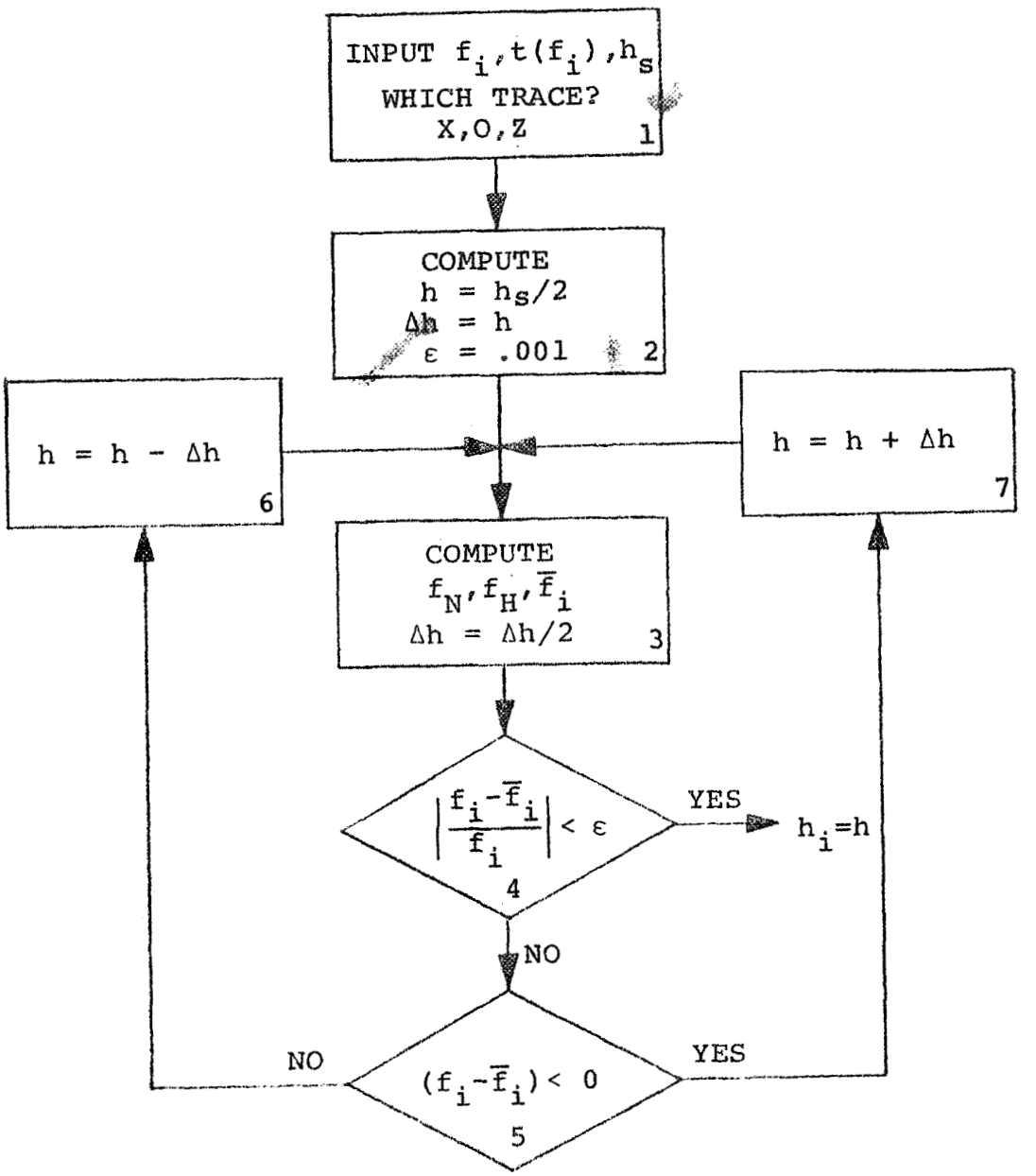


Fig. 4-4

Flow Diagram -

To Compute

h_i for Matrix Processing

5. If $(f_i - \overline{f}_i) < 0$

   NO, go to step 6.

   YES, go to step 7.

6. Compute next trial value, $h = h - \Delta h$

7. Compute next trial value, $h = h + \Delta h$

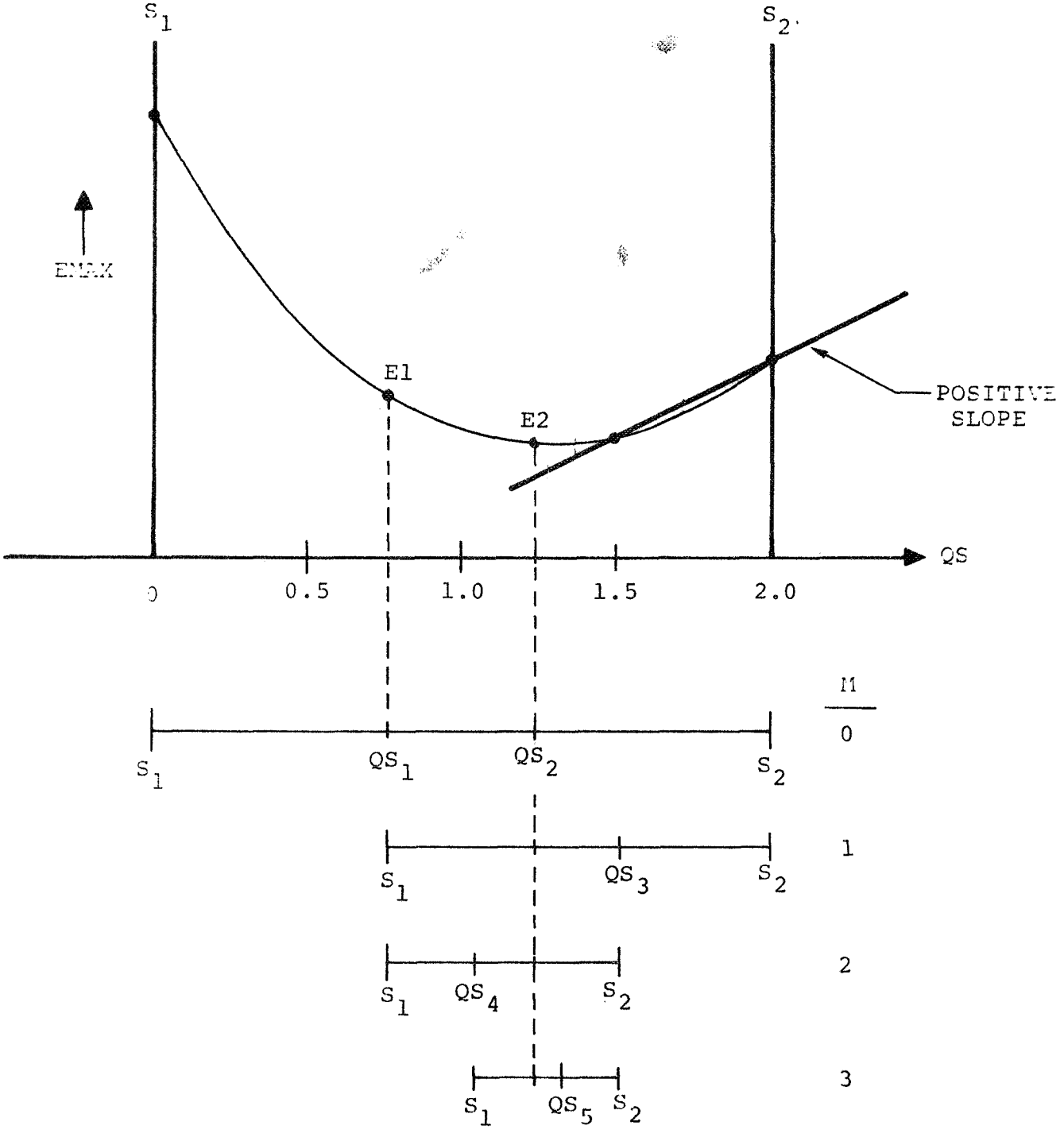

S_1
S_2
EMAX
E1
E2
POSITIVE SLOPE
QS
0
0.5
1.0
1.5
2.0
M
0
S_1
QS_1
QS_2
S_2
1
S_1
QS_3
S_2
2
S_1
QS_4
S_2
3
S_1
QS_5
S_2

FIGURE 4-5

4.4 PROGRAM NHMODEL

The Matrix method of reducing topside sounder ionograms to electron density profiles requires an initial estimate of N(h) as represented by a mathematical model. In other words, an initial set of values for $\underline{\alpha}$ are required. This is Step 3 in the flow diagram of Fig. 4-3.

While the program for the Matrix method is under development, it would be well to start with an initial function N(h, $\underline{\alpha}$) that is reasonably close to an actual N(h) profile computed by either Forward or Inverse Processing. Program NHMODEL was developed in order to curve fit a variety of different N(h) profiles during the investigation of ways to adequately model an N(h) profile. Some of the subroutines such as CALC and MXV can be used directly in the overall Matrix program. Other subroutines such as COST and STEP would be modified for use in the flow diagram of Fig. 4-3.

Program NHMODEL was developed using a time share computer terminal. The program is written mostly in FORTRAN IV with some variations that were permitted and others required by Tymshare FORTRAN IV. Differences in the variable names in the program and the nomenclature in paragraph 4.1 arise due to the fact that Program NHMODEL is a revision of a standard Astrodata program and it was convenient to minimize changes in the variable name structure.

A complete listing of Program NHMODEL is included starting on page IV-34. The numbers in the left hand column of each page are line numbers that are used in the time share mode of operation. They are convenient for reference but are not part of the actual program.

A flow diagram of program NHMODEL is included starting on page IV-33a. The results obtained from curve fitting the N(h) profiles from four different ionograms are shown in Tables 4-1 through 4-5.

Input data is set up in the DATA statements, lines 138 through 144.

NCP = number of complex poles

NRP = number of real poles

NRZ = number of zeros at the origin

The last two elements of array ZK are the scale factor ρ, and the change of variable constant η. Scale factor ρ is modified on each iteration at lines 580 and 824, but η remains constant throughout the program. In line 162, K is defined as the number of degrees of freedom in the $N(h_i, \underline{\alpha})$ model.

Array HN is a two-dimensional array which contains the input prescription vector, which is the $N(h_i)$ profile to be curve fit. The (HN(I,1), I = 1,KL) are values of true height in units of

MNEMONICS

NCP = Number of Complex Poles

NRP = Number of Real Poles

NRZ = Number of Zeros at Origin

ETA = Constant for Change of Variable $s = \eta - h$

ITCNT = Iteration Count

CK = ϵ in eqn. (4-33)

EMAX = Cost Functional, $|E(I)|_{max}$

QS = Step Size in eqn. (4-42)

J = Index for ZK Array

PZK(J) = Initial Pole Positions of Analytic Model

ZK(J) = Final Pole Positions of Analytic Model

I = Index of Input Prescription Vector HN

HN(I,1) = True Height in 10^3km

HN(I,2) = N (Electron Density x 10^{-4})

AMP(I) = Value of N from Analytic Model

E(I) = Curve Fit Error, $\ln[HN(I,2)/AMP(I)]$

Table 4-1. Mnemonics for Tables 4-2 through 4-5

```
/DATA5/      1-10-70      RUN 1      1345      /HN1/
↑
>RUN
NCP =     4     NRP =     2     NRZ =     1     ETA =     4

ITCNT     CK          EMAX          QS
 1        1     5.5517313E-02      1
```

J	PZK(J)	ZK(J)
1	.06000	.06837
2	3.60000	3.60745
3	1.30000	1.25722
4	2.90000	2.92058
5	1.90000	1.78564
6	3.90000	3.90132
7	4.50000	4.50581
8	1.50000	1.50520
9	4.20000	4.20497
10	.60000	.62368
11	1.00000	66444.57970
12	4.00000	4.00000

I	HN(I,1)	HN(I,2)	AMP(I)	E(I)
1	2.90000	.29200	.29086	.00392
2	2.80000	.30300	.30532	-.00764
3	2.70000	.31800	.32018	-.00684
4	2.60000	.33500	.33574	-.00221
5	2.50000	.35200	.35228	-.00079
6	2.40000	.37000	.37005	-.00014
7	2.30000	.39000	.38931	.00177
8	2.20000	.41100	.41030	.00170
9	2.10000	.43400	.43327	.00168
10	2.00000	.45900	.45846	.00117
11	1.80000	.51900	.51659	.00465
12	1.60000	.59000	.58715	.00484
13	1.40000	.67400	.67406	-.00010
14	1.20000	.79200	.78623	.00731
15	1.00000	.95600	.94811	.00829
16	.90000	1.06800	1.06727	.00069
17	.80000	1.22800	1.23776	-.00791
18	.70000	1.49600	1.50822	-.00813
19	.60000	2.00600	2.00623	-.00012
20	.50000	3.14700	3.15374	-.00214

```
PAUSE
320 >
```

Table 4-2. /HN1/ 65-340-003823 Alouette II Pass 81 at Ottawa

```
/DATA5/      1-13-70      RUN 2      1400      /HN4/
↑
240 >RUN
NCP =     6     NRP =     0     NRZ =     3     ETA =     5

ITCNT     CK          EMAX          QS
 1        1        .4405434      1
 2        1      6.816301E-02    1.0669977
 3       .3      4.699246E-02      .63314002
 4      9E-02     2.873801 3E-02     1.3839071
 5      2.7E-02     2.3954022E-02      .69599374
 6      1E-02     2.139765 8E-02      .38010141
 7      1E-02     1.992 9924E-02      .21983965
 8      1E-02     1.983 954E-02      .11457904
 9      1E-02     1.980 968E-02     7.2008353E-02
10      1E-02     1.979 544E-02     4.5704904E-02
11      1E-02     1.979 992E-02     2.3237761E-02

    J          P7K(J)          ZK(J)
    1          .06000          .05249
    2         7.00000         6.58393
    3          .09000          .06418
    4         6.11000         5.50347
    5          .05000          .05270
    6         4.69000         4.65728
    7          .30000          .42664
    8         4.70000         4.70640
    9          .50000         1.32791
   10         3.29000         2.99960
   11         1.02000          .68869
   12         1.75000         1.76945
   13         1.00000    124084.43000
   14         5.00000         5.00000

    I    HN(I,1)    HN(I,2)     AMP(I)      E(I)
    1    2.09966      .41211      .40895      .00771
    2    2.04912      .43946      .43731      .00491
    3    1.99390      .50675      .50802     -.00251
    4    1.73615      .67394      .67809     -.00628
    5    1.53350      .93711      .93364      .00371
    6    1.35295     1.28677     1.29261     -.00453
    7    1.16740     1.90108     1.91385     -.00669
    8    1.04249     2.64945     2.61095      .01464
    9     .94939     4.71941     4.68601      .00689
   10     .65850     9.96743    10.14657     -.01781
   11     .55883    17.92012    18.10745     -.01040
   12     .49919    28.10641    28.28757     -.00642
   13     .44505    41.50047    40.92579      .01394
   14     .41461    56.18123    55.56875      .01096
   15     .39580    74.68321    76.17590     -.01979
   16     .36753    92.61606    91.54081      .01168

STOP
(@320 )>
```

Table 4-3. /HN4/ 68-122-165346 Ionogram Recorded at Ottawa

```
/DATA5/      1-12-70      RUN 2      1120      /HN5/
↑
>RUN
NGF =     6     NPP =     0     NRZ =     3     ETA =     5

ITCNT     CK          EMAX          OS
 1        1      .46926132    1
 2        1      9.04999998E-02      1.2219699
 3       .3      4.35396998E-02      .6029999
 4      9E-02     3.3319191E-02       .4330292
 5     9.7E-02     2.6516545E-02      1.783138
 6      1E-02     2.3390153E-02       .7993103E
 7      1E-02     2.1799732E-02       .65187165
 8      1E-02     1.9990404E-02      1.6293207
 9      1E-02     1.9720951E-02       .56041912
10      1E-02     1.9703974E-02      -.13077974

   J          PZK(J)          ZK(J)
   1          .06000          .05518
   2         7.00000         6.45775
   3          .09000          .10759
   4         6.11440         5.63494
   5          .05000          .04754
   6         4.69191         4.67604
   7          .30000          .41342
   8         4.79133         4.67853
   9          .50019         1.
  10         3.29297         3.10759
  11         1.09749          .97991
  12         1.75346         1.76127
  13      99442.16000    130933.66100
  14         5.00000         5.00000

   I    HN(I,1)     HN(I,2)      AMP(I)       E(I)
   1    2.10120      .36963      .37621     -.01766
   2    2.06631      .41559      .40953      .01467
   3    1.91957      .52496      .51462      .01970
   4    1.66679      .75970      .77373     -.01961
   5    1.39598     1.25332     1.25970     -.00502
   6    1.10195     2.09720     2.07434      .01096
   7    1.04505     2.81831     2.79496      .01194
   8     .97041     4.65259     4.68493     -.00691
   9     .65622    10.59089    10.79145     -.01970
  10     .55197    19.99577    19.54694      .01492
  11     .45199    29.40599    29.06205      .01197
  12     .41493    43.33986    43.67107     -.00751
  13     .40625    55.25960    55.57974     -.00576
  14     .37113    73.99957    73.52974     -.00736
  15     .35991    94.73077    93.92319      .00856
  16     .33194   119.97670   119.60255     -.00273

 AUS
```

Table 4-4. /HN5/ 68-122-165408 Ionogram Recorded at Ottawa

```
 /DATA5/          1-13-70          RUN 5            1550       /HN6/
↑
>LIST 138:139
139             DATA NCP, NRP, NRZ /6, 0, 3/
139             DATA (ZK(K), K = 1,14) /.06,6.6,.07,5.5,
                                .056,4.7,.45,4.75,1.3,3.2,.85,1.7,1.,5./
>RUN
NCP =      6     NRP =      0     NRZ =      3     ETA =      5

ITCNT     CK          EMAX            QS
 1       1       .19462359    1
 2       1       2.6202655E-02       1.0896104
 3       .3      2.0441304E-02       .89979316
 4       9E-02     2.0446496E-02      -3.0656876E-03

     J           PZK(J)           ZK(J)
     1           .06000           .05984
     2          6.60000          6.62509
     3           .07000           .06633
     4          5.50000          5.50698
     5           .05600           .05992
     6          4.70000          4.68798
     7           .45000           .34404
     8          4.75000          4.82979
     9          1.30000          1.42056
    10          3.20000          3.27129
    11           .85000           .81550
    12          1.70000          1.64677
    13          1.00000     160412.94900
    14          5.00000          5.00000

     I    HN(I,1)    HN(I,2)     AMP(I)      E(I)
     1    2.16084      .33260      .33223      .00112
     2    2.06682      .38499      .38419      .00208
     3    1.86519      .52641      .52962     -.00608
     4    1.61567      .80906      .80973     -.00083
     5    1.38934     1.25600     1.24366      .00987
     6    1.19733     1.88959     1.88959     -.00000
     7    1.06615     2.63100     2.62495      .00230
     8     .92638     3.93651     3.93452      .00051
     9     .78902     6.28601     6.34915     -.00999
    10     .66704    10.62651    10.58890      .00355
    11     .56003    18.88184    18.62187      .01386
    12     .48175    28.96252    29.12638     -.00564
    13     .44155    42.40472    43.28067     -.02045
    14     .40827    57.29030    58.15866     -.01504
    15     .38508    75.75568    74.22366      .02043
    16     .36128    96.45132    95.30206      .01199
    17     .33612   119.93530   120.87486     -.01618
    18     .32266   131.76640   131.98951     -.00169
    19     .30911   139.13170   137.71923      .01020

STOP
(@320 )>
```

Table 4-5. /HN6/ 68-122-165450 Ionogram Recorded at Ottawa

10^3 km. The HN(I,2) are the related values of electron density divided by 10^4. KL is the number of elements in the prescription vector.

The program is initialized in lines 148 through 172. At lines 174 and 176, the ZK array is stored in PZK for comparison on final printout.

At lines 178 and 180, the change of variable computation takes place.

$$s_i = \eta - h_i \quad \text{from eqn. (4-2)}$$
$$i = 1, 2, \ldots, I$$

Subroutine CALC, which is first called at line 188, computes $N(s_i)$, eqn. (4-2), as AMP(I), using the values of $\underline{\alpha}$ in array ZK. In line 580, the scale factor ρ is computed initially so that in line 584

$$\text{AMP}(10) = \text{HN}(10,2) \tag{4-27}$$

This forces the two curves N(h) and $N(h_i, \underline{\alpha})$ to cross at the tenth lamination boundary.

In Subroutine COST, which is first called at line 196, the discrepancy vector E(I) is computed in line 816. This is similar to vector $\underline{E}$ in eqn. (4-10). In the same DO loop the sum set of

E(I) is accumulated so that the mean value can be computed in line 820. The mean value of E(I) is removed in line 830 which is equivalent to changing the scale factor ρ by a term called EXM in line 822. The scale factor correction takes place in line 824. On the last iteration, the scale factor correction is applied to AMP(I) in line 314 for printout. In the DO loop lines 828 through 836, the maximum absolute value of E(I) is defined as EMAX.

In Program NHMODEL, EMAX is used as the cost functional and the program is structured to minimize EMAX, although it minimizes all other values of E(I) as well. Beginning at line 838, if EMAX > .5 a different scale factor is computed as ER. This is used to scale down the discrepancy vector E(I) in line 844 so that the maximum value of $\underline{E}$ in the matrix equation does not exceed 0.5. This prevents a divergent condition from developing if the initial values of ZK are not very good.

Statement 100 at line 198 is the reentry point for the major iterative loop and is called at line 298. There are three exit conditions from this major loop at lines 202, 208, and 209.

The elements of the matrix of partial derivatives D(I,L) are computed in Subroutine GEND, which is called at line 216. These are similar to the elements a_{ik} of eqn. (4-9), but the

D(I,L) can be computed directly because N(h, $\underline{\alpha}$) is an analytic function of $\underline{\alpha}$ and is the model of the curve to be fitted. The elements are computed as the partial derivative of the natural logarithm of the function N(s, $\underline{\alpha}$) with respect to $\underline{\alpha}$ as given by

$$D(I,L) = \frac{\partial \ \ell n \ N(s_i)}{\partial \alpha_k} \tag{4-28}$$

where the notation D(I,L) is similar to a_{ik} in eqn. (4-9).

So far in the program we have developed a matrix equation

$$\underline{E} = \underline{D}(\underline{DK}) \tag{4-29}$$

where DK is the program name for the adjustment vector $\Delta\underline{\alpha}$. The similarity with eqn. (4-10) is obvious. Both sides of eqn. (4-29) are multiplied by $\underline{D}^T$ in lines 224 through 240. In the program

$$\underline{B} = \underline{D}^T\underline{E} \tag{4-30}$$

and

$$\underline{A} = \underline{D}^T\underline{D} \tag{4-31}$$

Since $\underline{A}$ is a symmetric matrix, only the terms on and above the main diagonal are computed directly. Terms below the main diagonal are filled in at line 254.

Under some conditions matrix $\underline{A}$ can be poorly conditioned so that it cannot be properly inverted. This condition is prevented by adding a constant CK to each term of the main diagonal at line 250. This

alters the matrix equation which is then compensated by adding the proper correction term to the other side of the equation. Assume a matrix equation of the form

$$\underline{A}\ \underline{X} = \underline{B} \tag{4-32}$$

It follows that

$$[\underline{A} + \varepsilon \underline{I}]\underline{X} = \underline{B} + \varepsilon \underline{X} \tag{4-33}$$

where I = identity matrix

In the program

$$\underline{DK} = \underline{X}$$

$$CK = \varepsilon$$

$$\underline{B1} = \underline{B} + \varepsilon \underline{X} \qquad \text{line 266}$$

Since $\underline{DK}$ is the unknown in the matrix equation, the first trial value of $\underline{DK}$ for eqn. (4-33) is the previous value from the prior iteration. $\underline{DK}$ is then computed in Subroutine MXV which gives a better value of $\underline{DK}$ for eqn. (4-33). This is repeated four times in lines 262 through 274, each time improving the value of DK. Under certain conditions CK is revised downward, lines 276 through 280, so that

$$.01 \leq CK \leq 1. \tag{4-34}$$

This reduces the product term $\varepsilon \underline{X}$ in eqn. (4-33) so that it becomes less and less significant in the value of $\underline{B1}$.

Subroutine MXV[24] computes $\underline{X}$ in eqn. (4-32) without evaluating $\underline{A}^{-1}$. For a 10 x 10 matrix this requires only about one half the computer CPU cycles compared with evaluating $\underline{X}$ using the equation

$$\underline{X} = \underline{A}^{-1}\ \underline{B} \tag{4-35}$$

MXV is modified from a generalized matrix inversion subroutine from McCracken.[20]

The magnitude of adjustment vector DK can be optimized to give maximum reduction in the cost functional EMAX. This is accomplished in Subroutine STEP using a Golden Section Search [26]. In the program, the step size QS becomes a scalar multiplier on DK in lines 928, 952, 966, and 994.

The boundaries of the search interval are S1 and S2, with initial values of 0 and 1 respectively in lines 918 and 920. In the first part of STEP, thru line 944, S2 is increased by increments of 0.5, line 936, until a line thru two consecutive values of EMAX has a positive slope, line 934. This establishes the minimum value of EMAX as lying somewhere between S1 and S2, as shown in Fig. 4-5.

The Golden Section search is based on setting up intervals with ratios such that

$$x^2 + x - 1 = 0 \tag{4-35}$$

$$x = \frac{\sqrt{5} - 1}{2} = .618$$

$$x^2 = .382$$

The search for minimum EMAX starts at line 948 with M = 0.

Compute the following:

$$\text{E1} = \text{EMAX}\Big|_{QS_1 = .382(S2 - S1)} \quad \text{lines 948-958} \tag{4-36}$$

$$\text{E2} = \text{EMAX}\Big|_{QS_2 = .618(S2 - S1)} \quad \text{lines 962-972} \tag{4-37}$$

Then compare to see which of E1 and E2 is larger, line 978.

$$\text{If} \quad \text{E2} < \text{E1} \tag{4-38}$$

$$\text{then} \quad QS_1 < QS < S2 \tag{4-39}$$

We reduce the search interval by making

$$S1 = QS_1 \qquad \text{line } 982 \tag{4-40}$$

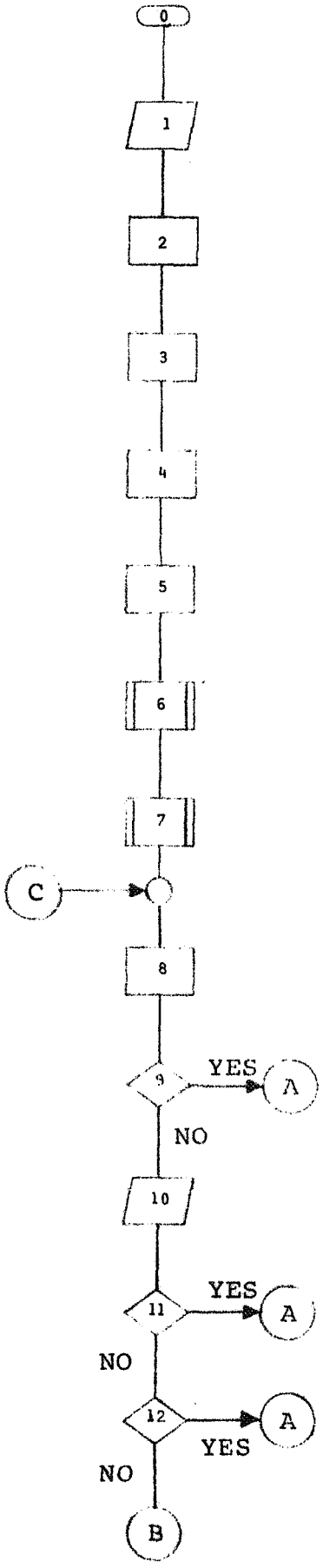
On the next iteration with M = 1, E1 = E2 and a new E2 is computed.

$$\text{E2} = \text{EMAX}\Big|_{QS_3 = S1 + .618(S2 - S1)} \quad \text{lines 962-972} \tag{4-41}$$

By using the ratios in the Golden Section search, it is only necessary to compute one new value of EMAX per iteration. After ten iterations, the step size QS for minimum EMAX has been determined. The loop exits at line 976 with M > 9. The scalar multiplication

$$DK(I) = DK(I)*QS$$
$$I = 1, 2, ..., K \qquad (4\text{-}42)$$

occurs at line 994, followed by a return to the main program. All the state variables should remain positive. If any elements become negative, they are forced positive at line 294. This completes the major iterative loop of Program NHMODEL. A detailed flow diagram for the Program NHMODEL is shown on Figures 4-6 through 4-11, and a program list is provided in Table 4-6.



Entry

1. Read input data NCP, NRP, NRZ, estimate of ZK vector, and density profile

2. Set CK = 1, QS = 1, DK vector = 0, and iteration count = 0

3. Calculate K, K1, and K2

4. Store the initial estimate of the ZK vector in PZK

5. Compute the change of variable
   S(I) = ZK(K2) - HN(I,1)

6. Compute the approximation to N(H)
   CALL CALC [AMP,HN,S,ZK]
   Return with AMP

7. Compute the discrepancy vector, E
   CALL COST [AMP,E,HN,ZK,EMAX,EXM]
   Return with E, EMAX, EXM

8. Increase iteration count by 1

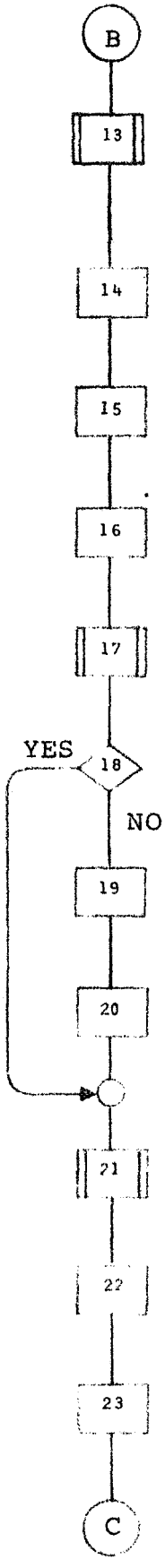
9. Is EMAX < 0.01?

10. Display iteration count, CK, EMAX, and QS

11. Is QS < 0.01?

12. Is CK = .01 and EMAX < .025 and QS < .03?

Fig. 4-6. Flow Diagram for Program NHMODEL



13. Compute partial derivative matrix, D
    CALL GEND[D,S,ZK]
    Return with D

14. Compute $\underline{B} = \underline{D}^T\underline{E}$

15. Compute $\underline{A} = \underline{D}^T\underline{D}$

16. Add CK to the main diagonal of the A matrix

17. Compute approximate solution to
    $\underline{DK} = (\underline{D}^T*\underline{D} + CK*\underline{I})^{-1}*(\underline{D}^T*\underline{E})$
    CALL MXV [A,DK,B1,T,K,K1]
    Return with DK

18. Is EMAX > 0.1?

19. Replace CK with 0.3 CK

20. If CK < 0.01, set CK = 0.01

21. Compute the multiplier, QS, to optimize the correction for the ZK vector. The new values for E and EMAX are computed.
    CALL STEP [AMP,DK,E,HN,S,ZK,EMAX,EXM,QS]
    Return with DK, QS

22. Replace ZK with ZK + QS*DK

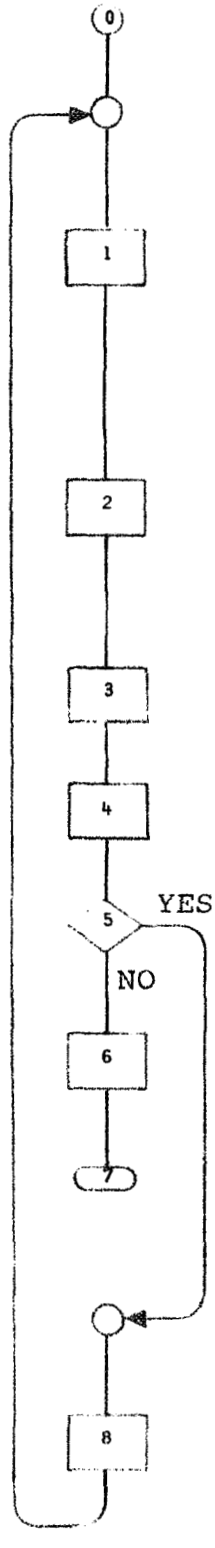
22. If any component of ZK is negative, then reverse the polarity of that component.

Fig. 4-6. Flow Diagram for Program NHMODEL (Cont)

24. Display HN(I,1),HN(I,2),AMP(I),E(I),
    I = 1 to KL

25. Exit

Fig. 4-6. Flow Diagram for Program NHMODEL (Cont)



Start

1. Compute the effect of the complex poles

$$DEN = \prod_{J=1}^{NCP} \left(\left| P_I \right| \left| P_I^* \right| \right)_J$$ (See Fig. 4-1)

2. Compute the effect of the real poles

Replace DEN with $DEN * \prod_{J=NCP+1}^{(NCP+NRP)} \left| P_I \right|_J$

3. Compute the effects of the zeros

$AMP(L) = [S(1)]^{NRZ}/DEN$

4. Compute scale factor ZK(K1) = HN(10,2)/AMP(10)

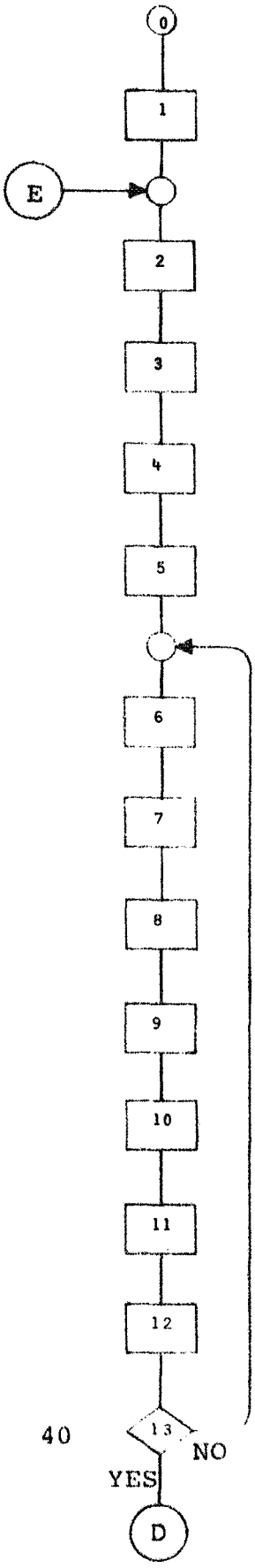
5. Is L = KL?

6. Replace AMP(L) with AMP(L)*ZK(K1), L = 1 to KL

7. Return

8. Increase L by 1

Fig. 4-7. NHMODEL Flow Diagram - Subroutine CALC [AMC,HN,S,ZK]



Start

1. Compute ZKP(I) = $[ZK(I)]^2$, I = 1 to K

2. DO 100 I = 1,K

3. Set L = 1

4. Compute S1 = $[S(I)]^2$

5. Compute S2 = S(I) + S(I)

6. DO 40 J = 1,NCP

7. Compute D1 = $[ZKP(L) + ZKP(L+1) - S1]^2$

8. Compute D2 = $[ZK(L) * S2]^2$

9. Compute RC = -D1 - D2

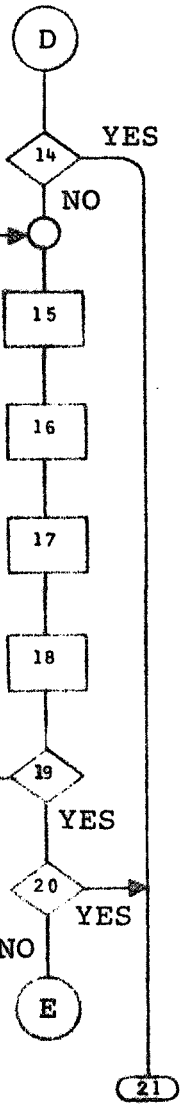
10. Compute D(I,L) = 2 * ZK(L) * [ZKP(L) + ZKP(L+1) + S1]/RC

11. Compute D(I,L+1) = 2 * ZK(L+1) * [ZKP(L) + ZKP(L+1) - S1]/RC

12. Increment L, L = L + 2

13. Is J = NCP in DO loop?

Fig. 4-8. NHMODEL Flow Diagram - Subroutine GEND [D,S,ZK]



14. Is NRP = 0

15. DO 60 J = 1,NRP

16. Compute RR = -S1 - ZKP(L)

17. Compute D(I,L) = ZK(L)/RR

18. Increment L, L = L + 1

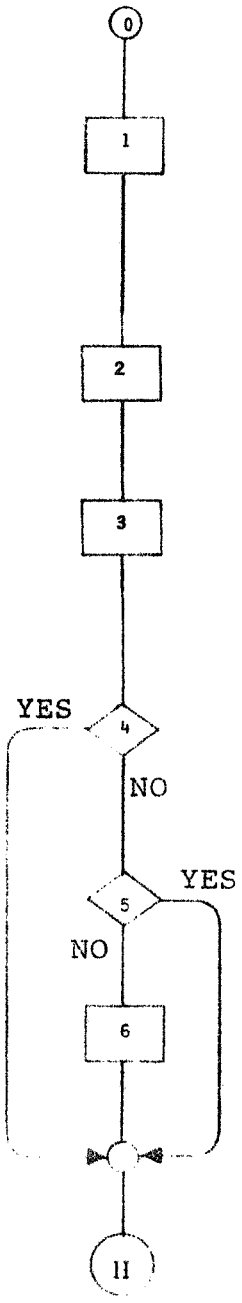
60 19. Is J = NRP in DO loop

100 20. Is I = KL

21. Return with

$$\begin{bmatrix} \frac{\partial \ell nAMP(1)}{\partial ZK(1)} & \frac{\partial \ell nAMP(1)}{\partial ZK(2)} & \cdots & \frac{\partial \ell nAMP(1)}{\partial ZK(K)} \\ \frac{\partial \ell nAMP(2)}{\partial ZK(1)} & \frac{\partial \ell nAMP(2)}{\partial ZK(2)} & \cdots & \frac{\partial \ell nAMP(2)}{\partial ZK(K)} \\ & \vdots & & \\ \frac{\partial \ell nAMP(KL)}{\partial ZK(1)} & \frac{\partial \ell nAMP(KL)}{\partial ZK(2)} & \cdots & \frac{\partial \ell nAMP(KL)}{\partial ZK(K)} \end{bmatrix}$$

Fig. 4-8. NHMODEL Flow Diagram - Subroutine GEND [D,S,ZK] (Cont)



Start

1. Load A matrix into array T

   $T(I,J) = A(I,J)$
   $I = 1, 2, \ldots, N$
   $J = 1, 2, \ldots, N$

2. Load vector B into last column of T

   $T(I,N1) = B(I),\ I = 1, 2, \ldots, N$

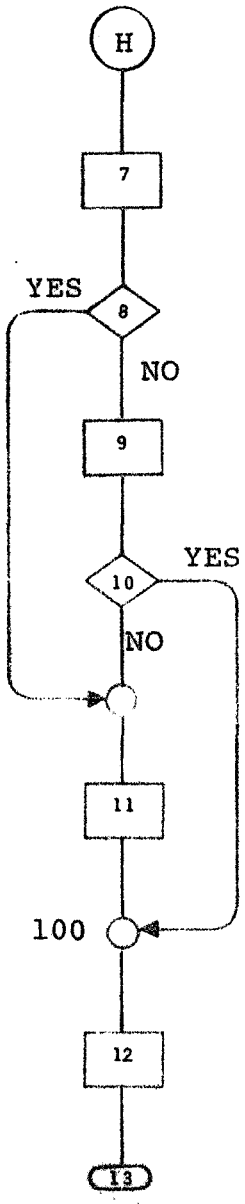
3. DO 100 for K = 1 to N
   In the k^{th} column, starting with the K+1 row, determine which row of the T matrix has the term with the greatest magnitude.
   This is indicated as the L^{th} row.

4. Is K = N?

5. Is L = K?

6. Interchange the L^{th} and K^{th} rows in the T matrix for all columns from the K^{th} through the $(N+1)^{th}$, J = K to N+1

Fig. 4-9. NHMODEL Flow Diagram - Subroutine MXV[A,X,B,T,N,N1]



7. Replace the T(K,J) term with T(K,J)/T(K,K) J = K+1 to N+1

8. Is K = 1?

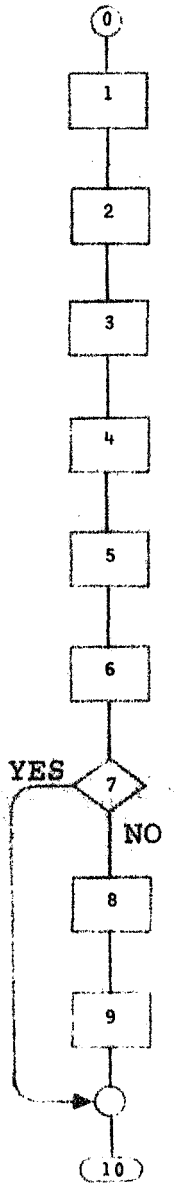
9. For the I^{th} row, from I = 1 to K-1, for J = K+1 to N+1, replace T(I,J) with T(I,J) - T(I,K)*T(K,J)

11. For the I^{th} row, from I = K+1 to N, for J = K+1 to N+1, replace T(I,J) with T(I,J) - T(I,K)*T(K,J)

12. X(I) = T(I,N1), for I = 1 to N

13. Return

Fig. 4-9. NHMODEL Flow Diagram - Subroutine MXV [A,X,B,T,N,N1] (Cont)



Start

1. Compute E(I) = ℓn[HN(I,2)/AMP(I)], I = 1 to KL

2. Compute $EMEAN = \frac{1}{KL} \sum_{I=1}^{KL} E(I)$

3. Compute $EXM = \varepsilon^{(EMEAN)}$

4. Replace ZK(K1) with ZK(K1)*EXM

5. Replace all E(I) with E(I) - EMEAN

6. Set EMAX = Max absolute value of E(I)

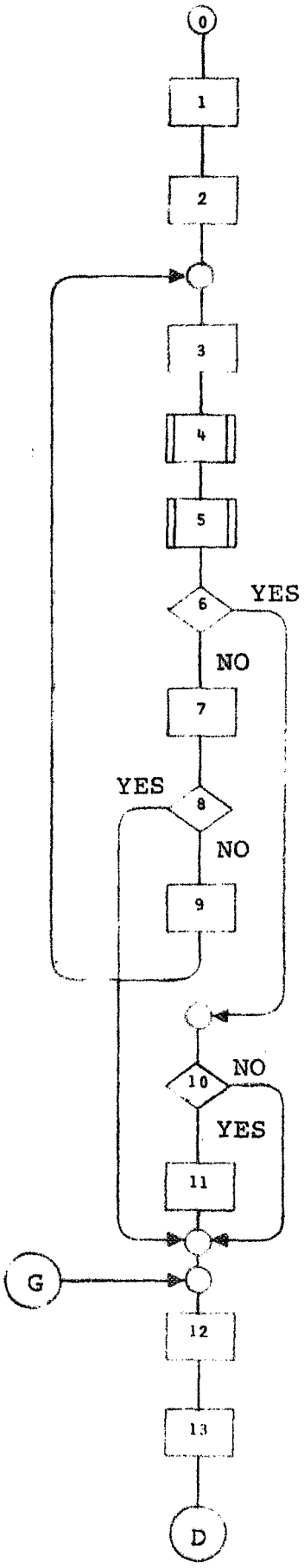
7. Is EMAX < 0.5?

8. Set ER = 0.5/EMAX

9. Replace all E(I) with E(I)*0.5/EMAX

10. Return

Fig. 4-10. NHMODEL Flow Diagram - Subroutine COST [AMP,E,HN,ZK,EMAX,EXM]



Start

1. Set E1 = EMAX, M = 0, S1 = 0, S2 = 1

2. Set QZK(I) = ZK(I), I = 1 to K

3. Replace ZK(I) with QZK(I) + DK(I)*S2, I = 1 to K

4. Call Subroutine CALC [AMP,HN,S,ZK]
   Return with AMP

5. Call Subroutine COST [AMP,E,HN,ZK,EMAX,EXM]
   Return with E, EMAX, EXM

6. Is EMAX > E1?

7. Increase S2 by 0.5

8. Is S2 > 2.5?

9. Set E1 = EMAX

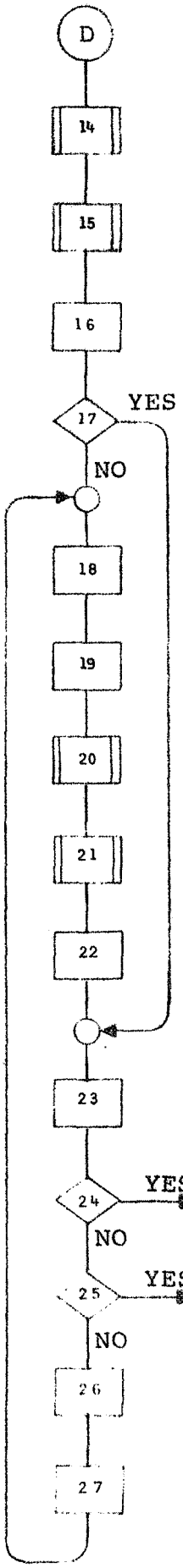
10. Is S2 = 1?

11. Set S1 = -1

12. Compute QS = S1 + (S2-S1)*0.382

13. Compute ZK(I) = QZK(I) + DK(I)*QS, I = 1 to 20

Fig. 4-11. NHMODEL Flow Diagram -
Subroutine STEP [AMP,DK,E,HN,S,ZK,EMAX,EXM,QS]



14. Call Subroutine CALC [AMP,HN,S,ZK]
    Return with AMP

15. Call Subroutine COST [AMP,E,HN,ZK,EMAX,EXM]
    Return with E, EMAX, EXM

16. Set E1 = EMAX

17. Is M > 0?

18. Compute QS = S1 + (S2-S1)*0.6]8

19. Compute ZK(I) = QZK(I) + DK(I)*QS, I = 1 to K

20. Call Subroutine CALC [AMP,HN,S,ZK]
    Return with AMP

21. Call Subroutine COST [AMP,E,HN,ZK,EMAX,EXM]
    Return with E, EMAX, EXM

22. Set E2 = EMAX

23. Increase M by 1

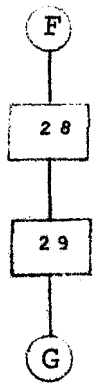
24. Is M > 9?

25. Is E2 $\geq$ E1?

26. Set E1 = E2

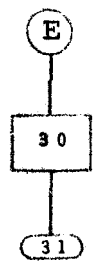
27. Replace S1 with S1 + (S2-S1)*0.382

Fig. 4-11. NHMODEL Flow Diagram -
Subroutine STEP [AMP,DK,E,HN,S,ZK,EMAX,EXM,QS] (Cont)



28. Set E2 = E1

29. Replace S2 with S1 + (S2-S1)*0.618



30. Replace DK(I) with DK(I)*QS, I = 1 to K

31. Return

Fig. 4-11. NHMODEL Flow Diagram -

Subroutine STEP [AMP,DK,E,HN,S,ZK,EMAS,EXM,QS] (Cont)

```
↑
LIST 1:199

100.        C:    PROGRAM NHMODEL
102.        C:
104.        C:    PROGRAM TO INITIALIZE THE ALPHA VECTOR
106.        C:    INPUT IS A SET OF N(H(I)) DATA POINTS IN ARRAY HN
108.        C:
110.        C:    ETA = ZK(K+2)
112.        C:     HN = PRESCRIPTION VECTOR (H,N)
114.        C:      K = NUMBER OF TERMS IN APPROXIMATION VECTOR ZK
116.        C:     KL = NUMBER OF TERMS IN PRESCRIPTION VECTOR HN
118.        C:    RHO = ZK(K+1)
120.        C:      S = ETA - HN(I,1)
122.        C:     ZK = ALPHA VECTOR
124.        C:
126.            DIMENSION TITLE (7), ZK(14), HN(20,2), S(20), DK(12)
128.            DIMENSION E(20), B(12), A(12,12), D(20,12), T(12,13)
130.            DIMENSION PZK(14), AMP(20), W(20), B1(12)
132.        C:
134.            COMMON NCP, NRP, NRZ, K, K1, KL
136.        C:
138.            DATA NCP, NRP, NRZ /6, 0, 3/
140.            DATA (ZK(K), K = 1,14) /.06,7.,.09,6.11,.05,4.68,.3,4.78,.
                50,3.29,1.02,1.75,1.,5./
142.            DATA KL, (HN(I,1), HN(I,2), I=1,8) /16, 2.09966,.41211, 2.
                04912,.43946, 1.93839,.50675, 1.73615,.67384, 1.53350,.937
                11, 1.35285,1.28677, 1.16740,1.90108, 1.04249,2.64945,/
144.            DATA (HN(I,1), HN(I,2), I=9,KL) /.84938,4.71841, .65850,9.
                96743,.55223,17.92012, .48818,28.10641, .44505,41.50047, .
                41461,56.18123, .38580,74.68321, .36753,92.61606/
146.        C:
148.            ITCNT = 0
150.            CK = 1.
152.            QS = 1.
154.            DO 70 I = 1,20
156.            E(I) = .01
158.            IF (I .LE. 12) DK(I) = 0.
160.        70  CONTINUE
162.            K = NCP+NCP+NRP
164.            K1 = K+1
166.            K2 = K+2
168.            DISPLAY "NCP =", NCP, "NRP =", NRP, "NRZ =", NRZ, "ETA =",
                 ZK(K2)
170.            DISPLAY " "
172.            DISPLAY "ITCNT    CK        EMAX          QS"
174.            DO 90 I = 1,K2
176.        90  PZK(I) = ZK(I)
178.            DO 90 I = 1,KL
180.        90  S(I) = ZK(K2) - HN(I,1)
182.        C:
184.        C:    COMPUTE APPROXIMATION TO N(H)
186.        C:
188.            CALL CALC [AMP,HN,S,ZK]
190.        C:
192.        C:    COMPUTE DISCREPANCY VECTOR E
194.        C:
196.            CALL COST [AMP,E,HN,ZK,EMAX,EXM]
198.       100  CONTINUE
```

Table 4-6. Program List for Program NHMODEL

```
↑
LIST 200:313

200.            ITCNT = ITCNT + 1
202.            IF (EMAX .LT. .01 ) GO TO 200
204.          C: DISPLAY "ZK =", ZK
206.            DISPLAY ITCNT, CK, EMAX, QS
208.            IF (QS .LT. .01) GO TO 200
209.            IF (CK .EQ. .01 .AND. EMAX .LT. .025 .AND. QS .LT. .03) GO
                 TO 200
210.          C:
212.          C:    GENERATE PARTIAL DERIVATIVE MATRIX D
214.          C:
216.            CALL GEND [D,S,ZK]
218.          C:
220.          C:    MULTIPLY BOTH SIDES BY D TRANSPOSE
222.          C:
224.            DO 120 I = 1,K
226.            B(I) = 0.
228.            DO 120 L = 1,KL
230.        120 B(I) = B(I) + D(L,I)*E(L)
232.            DO 130 I = 1,K
234.            DO 130 J = I,K
236.            A(I,J) = 0.
238.            DO 130 L = 1,KL
240.        130 A(I,J) = A(I,J) + D(L,I)*D(L,J)
242.          C:
244.          C:    ADD CK TO MAIN DIAGONAL AND COMPLETE THE A MATRIX
246.          C:
248.            DO 140 I = 1,K
250.            A(I,I) = A(I,I) + CK
252.            DO 140 J = 1,K
254.        140 A(J,I) = A(I,J)
256.          C:
258.          C:    COMPUTE ADJUSTMENT VECTOR DK ON ALPHA VECTOR ZK
260.          C:
262.            DO 160 J = 1,4
264.            DO 150 I = 1,K
266.        150 BI(I) = B(I) + DK(I)*CK
268.            CALL MXV [A,DK,BI,T,K,K1]
270.          C: DISPLAY " "
272.          C: DISPLAY "DK =", DK
274.        160 CONTINUE
276.            IF (EMAX .GT. .1) GO TO 170
278.            CK = CK*0.3
280.            IF (CK .LT. .01) CK = .01
282.          C:
284.          C:    COMPUTE STEP SIZE FOR ADJUSTMENT VECTOR DK AND ADD D
                 K TO ALPHA VECTOR ZK
286.          C:
288.        170 CALL STEP [AMP,DK,E,HN,S,ZK,EMAX,EXM,QS]
290.            DO 190 I = 1,K
294.            IF (ZK(I) .LT. 0.) ZK(I) = -ZK(I)
296.        190 CONTINUE
297.            GO TO 100
300.        200 CONTINUE
302.            DISPLAY " "
304.            DISPLAY "    J          PZK              ZK"
306.            WRITE (1,921) (J, PZK(J), ZK(J), J = 1,K2)
308.            DISPLAY " "
310.            DISPLAY "    I   HN(I,1)   HN(I,2)    AMP(I)     E(I)"
312.            DO 220 I = 1,KL
```

Table 4-6. Program List for Program NHMODEL (Cont) IV-35

```
PROGRAM LIST      NHMODEL      1-13-70      PAGE 3
↑
LIST 313:499

314.            AMP(I) = AMP(I)*EXM
316.            WRITE (1,919) I, HN(I,1), HN(I,2), AMP(I), E(I)
318.        220 CONTINUE
320.            STOP
322.        900 FORMAT (7A10)
324.        901 FORMAT (8I5)
326.        902 FORMAT (I5, 2F10.5)
328.        903 FORMAT (I5, F10.5, E10.5)
330.        919 FORMAT (I5, 4F10.5)
332.        920 FORMAT (I5, 3F10.5)
334.        921 FORMAT (I5, 2F15.5)
336.            END

>
```

Table 4-6. Program List for Program NHMODEL (Cont)

```
PROGRAM LIST     NHMODEL      1-13-70      PAGE 4
↑
LIST 500:599

500.          SUBROUTINE CALC [AMP,HN,S,ZK]
502.       C:
504.       C:    CALCULATE N(S) OF THE APPROXIMATING FUNCTION ZK
506.       C:
508.          DIMENSION AMP(20), HN(20,2), S(20), ZK(14)
510.          COMMON NCP, NRP, NRZ, K, KI, KL
512.       C:
514.       C: DISPLAY "   I", "       Q1", "          DEN"
516.          DO 50 L = 1,KL
518.          I = 1
520.          DEN = 1.0
522.       C:
524.       C:    COMPUTE AMPLITUDE RESPONSE DUE TO COMPLEX POLES
526.       C:
528.          DO 20 J = 1,NCP
530.          A1 = ZK(I)
532.          B1 = ZK(I+1)
534.          R1 = (B1-S(L))*(B1-S(L))
536.          R2 = (B1+S(L))*(B1+S(L))
538.          R3 = A1*A1
540.          Q1 = SQRT[(R1+R3)*(R2+R3)]
542.          DEN = DEN * Q1
544.       C: DISPLAY I, Q1, DEN
546.       20 I = I + 2
548.       C:
550.       C:    COMPUTE AMPLITUDE RESPONSE DUE TO REAL POLES
552.       C:
554.          IF (NRP .EQ. 0) GO TO 40
556.          DO 30 J = 1,NRP
558.          A1 = ZK(I)
560.          R1 = S(L)*S(L)
562.          R2 = A1*A1
564.          Q1 = SQRT[R1+R2]
566.          DEN = DEN * Q1
568.       C: DISPLAY I, Q1, DEN
570.       30 I = I + 1
572.       40 CONTINUE
574.          AMP(L) = S(L)**NRZ/DEN
576.       C: DISPLAY I, Q1, DEN, AMP(L), L
578.       50 CONTINUE
580.          ZK(KI) = HN(10,2)/AMP(10)
582.          DO 60 L = 1,KL
584.       60 AMP(L) = AMP(L)*ZK(KI)
586.          RETURN
588.          END

>
```

Table 4-6. Program List for Program NHMODEL (Cont)

```
↑
LIST 600:699

600.          SUBROUTINE GEND (D,S,ZK)
602.        C:
604.        C:    GENERATES D MATRIX OF PARTIAL DERIVATIVES
606.        C:
608.          DIMENSION S(20), ZK(14), ZKP(12), D(20,12)
610.          COMMON NCP, NRP, NRZ, K, K1, KL
612.        C:
614.          DO 10 I = 1,K
616.       10 ZKP(I) = ZK(I)*ZK(I)
618.          DO 100 I = 1,KL
620.          L = 1
622.          S1 = S(I)*S(I)
624.          S2 = S(I) + S(I)
626.        C:
628.        C:    TERMS FOR COMPLEX POLES
630.        C:
632.          DO 40 J = 1,NCP
634.          D1 = (ZKP(L)+ZKP(L+1)-S1)**2
636.          D2 = (ZK(L)*S2)**2
638.          RC = -D1-D2
640.          D(I,L)   = (2.*ZK(L)  *(ZKP(L)+ZKP(L+1)+S1))/RC
642.          D(I,L+1) = (2.*ZK(L+1)*(ZKP(L)+ZKP(L+1)-S1))/RC
644.       40 L = L + 2
646.        C:
648.        C:    TERMS FOR REAL POLES
650.        C:
652.          IF (NRP .EQ. 0) GO TO 100
654.          DO 60 J = 1,NRP
656.          RR = -S1-ZKP(L)
658.          D(I,L) = ZK(L)/RR
660.       60 L = L + 1
662.      100 CONTINUE
664.          RETURN
666.          END

>
```

Table 4-6. Program List for Program NHMODEL (Cont)

```
↑
>LIST 700:799


700             SUBROUTINE MXV (A, X, B, T, N, N1)
702             C:    SIMPLIFIED MATRIX INVERSE SOLUTION PROGRAM
704             C:    GIVEN A * X = B FIND X = A**-1 * B
706             C:    LOAD A(N,N) IN FIRST N COLUMNS OF T(N,N1)
708             C:         B IN LAST COLUMN
710             C:    AFTER GAUSS-JORDAN PROCESS X WILL APPEAR IN LAST COL
                UMN OF T.
712             C:
714             DIMENSION A(12,12), X(12), B(12), T(12,13)
716             C:
718             DO 10 I = 1,N
720             DO 10 J = 1,N
722     10      T(I,J) = A(I,J)
724             DO 20 I = 1,N
726     20      T(I,N1) = B(I)
728             DO 100 K = 1,N
730             K1 = K + 1
732             IF (K .EQ. N) GO TO 50
734             L = K
736             DO 30 I = K1,N
738             IF (ABS(T(I,K)) .GT. ABS(T(L,K))) L = I
740     30      CONTINUE
742             IF (L .EQ. K) GO TO 50
744             DO 40 J = K,N1
746             Q = T(K,J)
748             T(K,J) = T(L,J)
750     40      T(L,J) = Q
752     50      CONTINUE
754             DO 60 J = K1,N1
756     60      T(K,J) = T(K,J)/T(K,K)
758             IF (K .EQ. 1) GO TO 80
760             KM1 = K - 1
762             DO 70 I = 1,KM1
764             DO 70 J = K1,N1
766     70      T(I,J) = T(I,J) - T(I,K) * T(K,J)
768             IF (K .EQ. N) GO TO 100
770     80      CONTINUE
772             DO 90 I = K1,N
774             DO 90 J = K1,N1
776     90      T(I,J) = T(I,J) - T(I,K) * T(K,J)
778     100     CONTINUE
780             DO 110 I = 1,N
782     110     X(I) = T(I,N1)
784             RETURN
786             END
>
```

Table 4-6. Program List for Program NHMODEL (Cont)

```
PROGRAM LIST      NHMODEL      1-13-70      PAGE 7
↑
LIST 800:899


802.          SUBROUTINE COST [AMP,E,HN,ZK,EMAX,EXM]
804.       C:
806.          DIMENSION AMP(20), E(20), HN(20,2), ZK(14)
808.          COMMON NCP, NRP, NRZ, K, K1, KL
810.       C:
812.          SUM = 0.
814.          DO 10 I = 1,KL
816.          E(I) = ALOG[HN(I,2)/AMP(I)]
818.       10 SUM = SUM + E(I)
820.          EMEAN = SUM/KL
822.          EXM = EXP[EMEAN]
824.          ZK(K1) = ZK(K1)*EXM
826.          EMAX = 0.
828.          DO 20 I = 1,KL
830.          E(I) = E(I) - EMEAN
832.          Q1 = ABS[E(I)]
834.          IF (EMAX .LT. Q1) EMAX = Q1
836.       20 CONTINUE
838.          IF (EMAX .LT. .5) GO TO 40
840.          ER = .5/EMAX
842.          DO 30 I = 1,KL
844.       30 E(I) = E(I)*ER
846.       40 RETURN
848.          END

>
```

Table 4-6. Program List for Program NHMODEL (Cont)

```
↑
LIST 900:999


      SUBROUTINE STEP [AMP,DK,E,HN,S,ZK,EMAX,EXM,QS]
C*
      DIMENSION AMP(20), E(20), HN(20,2), S(20), ZK(14)
      DIMENSION DK(12), QZK(12)
      COMMON NCP, NRP, NRZ, K, K1, KL
C*
      E1 = EMAX
      M  = 0
      S1 = 0.
      S2 = 1.
      DO 10 I = 1,K
   10 QZK(I) = ZK(I)
   20 DO 30 I = 1,K
   30 ZK(I) = QZK(I) + DK(I)*S2
      CALL CALC [AMP,HN,S,ZK]
      CALL COST [AMP,E,HN,ZK,EMAX,EXM]
      IF ((EMAX-E1) .GT. 0.) GO TO 40
      S2 = S2 + 0.5
      IF (S2 .GT. 2.5) GO TO 100
      E1 = EMAX
      GO TO 20
   40 IF (S2 .EQ. 1.) S1 = -1.
C*
  100 QS = S1 + (S2-S1)*.382
      DO 120 I = 1,K
  120 ZK(I) = QZK(I) + DK(I)*QS
      CALL CALC [AMP,HN,S,ZK]
      CALL COST [AMP,E,HN,ZK,EMAX,EXM]
      E1 = EMAX
      IF (M .GT. 0) GO TO 160
  130 QS = S1 + (S2-S1)*.618
  140 DO 150 I = 1,K
  150 ZK(I) = QZK(I) + DK(I)*QS
      CALL CALC [AMP,HN,S,ZK]
      CALL COST [AMP,E,HN,ZK,EMAX,EXM]
      E2 = EMAX
  160 M = M + 1
      IF (M .GT. 9) GO TO 200
      IF (E2-E1) 170,180,180
  170 E1 = E2
      S1 = S1 + (S2-S1)*.382
      GO TO 130
  180 E2 = E1
      S2 = S1 + (S2-S1)*.618
      GO TO 100
  200 DO 210 I = 1,K
  210 DK(I) = DK(I)*QS
      RETURN
      END
```

Table 4-6. Program List for Program NHMODEL (Cont)

4.5 THE PSEUDOINVERSE

Kinkel[28] has developed an efficient algorithm for computing the pseudoinverse of a matrix which can simplify the solution of eqn. (4-32).

The weighted least squares solution for $\underline{X}$ using the pseudoinverse is represented as follows:

$$\underline{X} = [A^T W A]^+ A^T W \underline{B} \qquad (4\text{-}42)$$

Even if A is poorly conditioned so that it cannot be inverted, the weighted least squares solution of $\underline{X}$ is given by eqn. (4-42).

The use of the pseudoinverse will avoid the iterative method associated with eqn. (4-33).

SECTION V

HORIZONTAL PROCESSING

5.0 INTRODUCTION

Present methods for routine topside ionogram data reduction depend on the following simplifying assumptions:

a. Vertical propagation of the echo pulse
b. Spherically stratified ionosphere over the period of the X-Trace

It is a basic property of the topside sounding satellites using swept frequency radar techniques that successive h'(f) echo pulses correspond to ionospheric soundings at different earth-based coordinates. It is well known that the ionosphere is nonspherically stratified[14] and since the Alouette II satellite moves at an average velocity of approximately 7 km/sec the effects of horizontal gradients in the orbital plane should be taken into consideration.

Present ionogram data reduction methods for h'(f) $\rightarrow$ N(h) computation assume a constant electron density profile for the ionosphere during the period of useful data acquisition in each ionogram. In addition, the Forward, Inverse, and Matrix processing methods are concerned with reducing the data from only one

ionogram at a time. In the Forward and Inverse methods, data reduction begins at satellite height and works downward as laminations are added to make up the N(h) profile. This has become known as Vertical processing.

The method of Horizontal Processing described in this section is proposed as a method of providing first order correction of the effects of horizontal gradients. The method assumes a constant horizontal gradient between adjacent ionograms in the same pass at equal true height lamination boundaries.

In this section the horizontal gradient is defined as the rate of change of electron density with respect to distance in the orbital plane at constant true height.

This can be represented as follows:

$$\left(\frac{\partial N}{\partial x}\right)_h \qquad (5\text{-}1)$$

where N = density in electrons/cm^3

x = distance in orbital plane

h = true height in 10^3km

Since time is one of the parameters associated with an ionogram, it is more convenient in data reduction to work with the gradient as a function of time than with distance. The gradient with respect to time is as follows:

$$\left(\frac{\partial N}{\partial t}\right)_h = \left(\frac{\partial N}{\partial x}\right)_h \left(\frac{\partial x}{\partial t}\right)_h \qquad (5\text{-}2)$$

At any true height level, the term $\left(\frac{\partial x}{\partial t}\right)_h$ is essentially constant for the period of one ionogram. For Horizontal Processing the gradient in eqn. (5-2) is assumed constant between adjacent ionograms at each lamination boundary.

Representative N(h,t) profiles for two ionograms are shown in Fig. 5-1 referred to a time scale. The N(h) profile shown as curve AB is the intersection of the ionospheric density surface ABGH and constant time plane t_{11}. The N(h,t) profile of Ionogram 1 is shown as curve AC which is the intersection of ABGH and a skewed time surface. This is the type of electron density profile that will be computed with the Horizontal Processing method. Similarly the N(h,t) profile of Ionogram 2 is shown as curve EG.

With Horizontal Processing it is necessary to work with a minimum of two adjacent ionograms from the same pass. The horizontal gradient is then included in the N(h,t) computation by linear

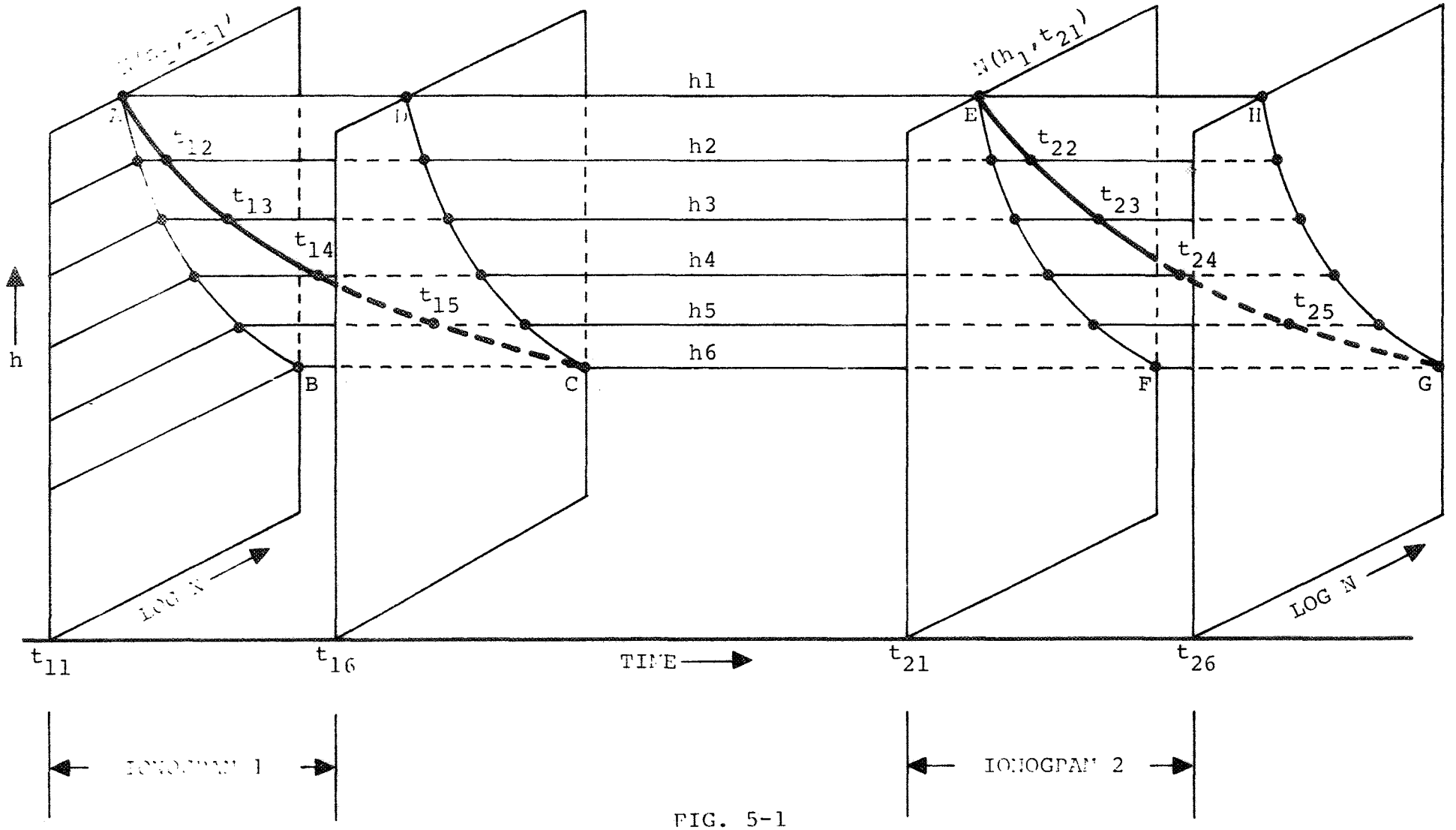

h1
h2
h3
h4
h5
h6
h
A
B
C
D
E
F
G
H
t_{12}
t_{13}
t_{14}
t_{15}
t_{22}
t_{23}
t_{24}
t_{25}
LOG N
LOG N
t_{11}
t_{16}
TIME
t_{21}
t_{26}
IONOGRAM 1
IONOGRAM 2

FIG. 5-1

interpolation of N with respect to time along each lamination boundary. This condition requires that the lamination boundaries be at the same time height for each ionogram. Horizontal Processing is most easily implemented by using Inverse Processing because the heights of the lamination boundaries will be the same for all ionograms in a pass, except at satellite height. We can also assume that $\partial N/\partial t$ is a constant along the satellite track so that $N(h_s,t)$ can be obtained by linear interpolation.

5.1 HORIZONTAL INVERSE PROCESSING

The method of Horizontal Processing will be developed as an extension of Inverse Processing, which is described in Section III. X-Trace data from two adjacent ionograms is scaled in conventional manner, and then curve fit so that $h'_x(f)$ is defined for any frequency on both X-Traces.

The value of $N(h_1,t)$ at satellite height is computed for each ionogram using the relation

$$f_N = \sqrt{f_1(f_1 - f_H)} \qquad (5\text{-}3)$$

where $f_1 = f_{xs}$

then $$N(h_1,t) = 12400\ f_N^2 \qquad (5\text{-}4)$$

Referring to Fig. 5-1, the electron density at the satellite for Ionogram 1 is denoted by $N(h_1,t_{11})$ and for Ionogram 2 by $N(h_1,t_{21})$. The value of $N(h_1,t)$ at any time between t_{11} and t_{21} is then determined by linear interpolation. The gradient along the satellite track is approximated by

$$S_1 = \frac{\Delta N}{\Delta t} = \frac{N(h_1,t_{21}) - N(h_1,t_{11})}{t_{21} - t_{11}} \tag{5-5}$$

Then at any time t along the satellite track from t_{11} to t_{16}

$$N(h_1,t) = N(h_1,t_{11}) + S_1(t - t_{11}) \tag{5-6}$$

It is convenient to denote satellite height by h_1 and yet, because of the elliptical satellite orbit, h_1 will change from one ionogram to the next. This doesn't change the validity of eqn. (5-6), but the change in height should be taken into account when computing the gyrofrequency f_H for successive laminations.

The bottom of lamination 1 is denoted by h_2 in Fig. 5-1. For Ionogram 1 the first trial value of $N(h_2,t_{12})$ is given by eqn. (3-16) which is repeated for convenience.

$$N_2 = N_1 e^{\left(\frac{h_2-h_1}{a_2}\right)} \tag{3-16}$$

where $N_1 = N(h_1, t_{11})$

a_2 is derived empirically

From this a value $\overline{h'_x(f_{21})}$ as in Fig. 3-2 is computed by Reverse Processing. We now have a trial value for t_{12} in Fig. 5-1 given by

$$t_{12} = t(f_{21}) \tag{5-7}$$

A value for $N(h_1, t_{12})$ is now computed with eqn. (5-6), using $t = t_{12}$. For the second iteration choose $N(h_2, t_{12})$ using eqn. (3-18)

$$N_{j2} = N_{j1}(1 \pm \alpha) \tag{3-18}$$

and compute a second point $\overline{h'_x(f_{22})}$ as in Fig. 3-2. A straight line thru points $\overline{h'_x(f_{21})}$ and $\overline{h'_x(f_{22})}$ will intersect the X-Trace at frequency f_{23}. Next, update the estimate for t_{12} by

$$t_{12} = t(f_{23}) \tag{5-8}$$

The value of $N(h_1, t_{12})$ is again updated using $t = t_{12}$ in eqn. (5-6). For the third iteration,

$$f_{N2} = \sqrt{f_{23}(f_{23} - f_{H2})} \tag{3-19}$$

and a third point $\overline{h'_x(f_{33})}$ is computed. A segment of a circle passing thru the three computed values of $h'_x(f_2)$ will intersect

the X-Trace at frequency f_{24}. Then as described on page III-9, the final value of $N(h_2,t_{12})$ can be computed from eqn. (3-19), (3-20) with $f_j = f_{24}$.

The corresponding point $N(h_2,t_{22})$ is computed for Ionogram 2 in a similar manner except that $N(h_1,t_{22})$ is computed by extrapolation. Assume the same slope as given by eqn. (5-5), then

$$N(h_1,t_{22}) = N(h_1,t_{21}) + S_1(t_{22} - t_{21}) \qquad (5\text{-}9)$$

The horizontal gradient for all other laminations is approximated by

$$S_j = \frac{N(h_j,t_{2j}) - N(h_j,t_{1j})}{t_{2j} - t_{1j}} \qquad (5\text{-}10)$$

where $j = 2, 3, \ldots, J$ lamination boundary number

In computing the values of $N(h_j,t_{1j})$ and $N(h_j,t_{2j})$ for succeeding laminations, the significant difference in Horizontal Processing is that the delay in previous laminations changes with the interpolated values of $N(h_i,t_{1i})$ and the extrapolated values of $N(h_i,t_{2i})$ where $i = 1, 2, \ldots, j$. In Vertical Processing it is assumed that the N(h) profile from previous laminations remains constant. From the example shown in Fig. 5-1, it can be seen

that correction for the effects of horizontal gradients requires the use of extrapolated data in Ionogram 2, while for Ionogram 1 the correction is obtained by interpolation. The use of extrapolated data values can be minimized by processing the ionograms in a pass starting with the last two ionograms and working in reverse sequence. After the last two ionograms have been reduced this way, the one referred to as Ionogram 1 becomes the interpolation reference for the next previous ionogram and no further extrapolation is required for the remaining ionograms in the pass.

The final output N(h,t) profile is then properly the intersection of the ionospheric density surface ABGH of Fig. 5-1 and a constant time plane. It then is a matter of choosing the time plane in each ionogram at which to compute N(h,t). In Fig. 5-1, the curve AB is the N(h,t) profile where $t_{11} = t(f_{xs})$. One may wish to select the time plane at $t = t_{16}$. This is convenient in that the curve DC is already known at the time the value of N_j is computed at the lower boundary of the last lamination.

5.2 HORIZONTAL MATRIX PROCESSING

Horizontal processing using the Matrix method of computing the N(h,t) profile can be implemented by setting up the analytic model of N(h,t) to converge to curve AC of Fig. 5-1. This will then provide a direct mapping relationship as shown in Fig. 4-2, since the variables in both planes are functions of time.

As a starting point, it may be desirable to compute curves AC and EG of Fig. 5-1 using the method of Section 5.1. Then make adjustments on AC using the matrix method of Section IV with curve EG as an interpolation reference for accommodating the horizontal gradient. When a matrix solution of curve AC is obtained, it then becomes the interpolation reference for a matrix solution of curve EG.

A different starting point would be to compute N(h) for ionograms 1 and 2 using the Inverse Mixed-Mode Processing method of Section 3.3. The computation for Horizontal Matrix Processing would then continue as described above.

Interpolation of N(h,t) for horizontal gradient compensation would proceed as in Section 5.1, where reference values for interpolation along curve AC would be computed with respect to the analytic model along lamination boundaries at true height levels computed as in Section 4.3

Prepared for:

Astrodata, Inc.
240 E. Palais Rd.
Anaheim, Calif.

Astrodata Project 938400

TR-1

SOME PROPOSED METHODS FOR REDUCTION OF TOPSIDE IONOGRAMS TO ELECTRON DENSITY PROFILES

September 25, 1969

John F. Kinkel
Technical Consultant
Newport Beach, Calif.

TABLE OF CONTENTS

Abstract

This paper proposes a method which may be used to find an electron density profile yielding the least squares error fit to the observations. Operator judgment of the quality of observation is taken into account in the data reduction process. A method by which time and position variations occurring over the observation period may be corrected for is also proposed.

Introduction

Some basic techniques for reduction of ionogram data have been described by Jackson[13]. Improved processing techniques have been investigated by Madsen[16]. The basic problem is to find the electron density, N, as a function of altitude, h, corresponding to the observation of apparent path length, h', as a function of frequency, f. The relationship between N(h) and h'(f) is such that it is convenient to map a set of points from the N(h) plane to the h'(f) plane and inconvenient to map from h'(f) to N(h). For this reason iterative procedures have been used to find N(h) given h'(f). Madsen terms the mapping from N(h) to h'(f) the Inverse mapping. Madsen's inverse mapping technique requires interpolation between observation points in the h'(f) plane. Neither Jackson's nor Madsen's techniques consider the time/position variation of the observation over the duration of a single ionogram.

Madsen has suggested** that present methods of processing ionogram data might be improved by a method incorporating either or both of the following features:

---

* Numbers in parenthesis refer to bibliography.

** Verbal discussion.

1. Time/position variation over the observation period of one ionogram is accounted for.

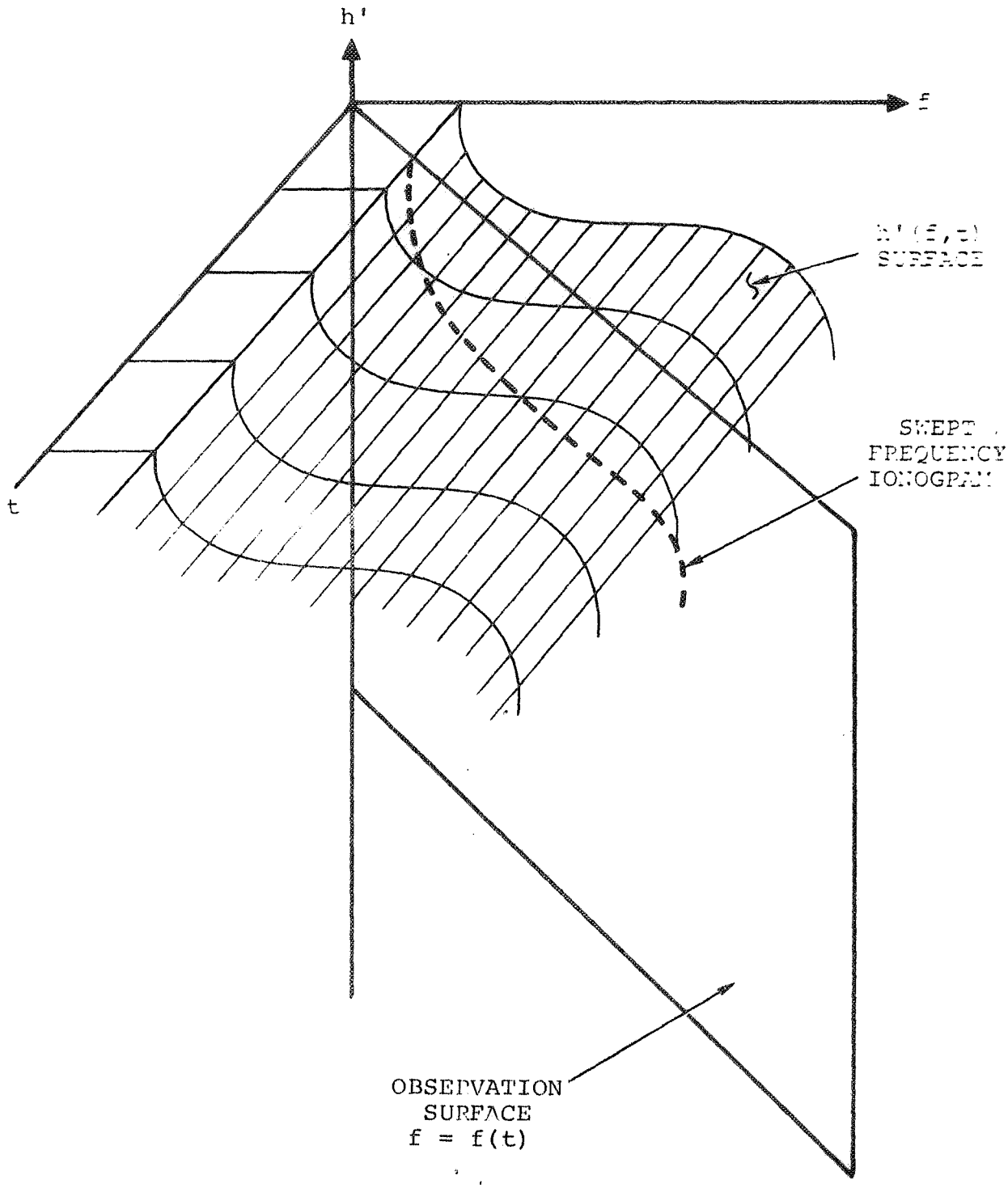
2. Data from usable portions of all three traces, Z, O, and X, from a single ionogram are used effectively to generate the N(h) profile.

The balance of this paper describes proposed methods for accomplishing these goals. The methods described for accomplishing items 1 and 2 above are essentially independent; i.e., either may be used without the other.

Approximations of h'(f,t)

A basic problem with swept frequency ionogram data is that h'(f) is also a function of time (and hence position) whereas what we would like is h'(f) data for a fixed time and position. Therefore, we should write our observation as h'(f,t) rather than h'(f) to show the time dependence. This is illustrated in Fig. TR-1-1 in which we see that h'(f,t) forms a surface in the three dimensional cartesian coordinate system (h',f,t). All observations made using a swept frequency sounding system must lie in the observation surface, f(t). Therefore, as shown in Fig. TR-1-1, a swept frequency ionogram consists of the locus of points forming the intersection of the two surfaces h'(f,t) and f(t). A sequence of ionograms can be visualized by a sequence of observation surfaces spaced along the t axis in Fig. TR-1-1.

While it is true that the time/position dependence of the electron density profile, N(h,t), could possibly be found directly from h'(f,t), it appears far simpler to first find $h'(f,t_k)$, where t_k is some particular point in time, and from this find $N(h,t_k)$. With this approach the mapping (or inverse mapping) problem is not increased in complexity by the added dimension of time/position.


h'
f
t
h'(f,t)
SURFACE
SWEPT
FREQUENCY
IONOGRAM
OBSERVATION
SURFACE
f = f(t)

Basically, our problem, then, is to construct the surface $h'(f,t)$ of Fig. TR-1-1 based on our $h'(f(t))$ observation. The intersection of this $h'(f,t)$ surface with the plane, $t = t_k$, yields $h'(f,t_k)$, the desired result.

A curve may be fit to a set of points by a number of common methods including Lagrange polynomial interpolation[6], spline fit[3], and an approximating function[3]. Both the Lagrange polynomial interpolation and the spline fit pass exactly through all of the given points whereas the approximating function will not. Therefore, for a large number of noisy observations the approximating function method may give superior results. Birkhoff and Garabedian[4] have described a method of spline fit for a surface. However, because of the large number of noisy observations within an ionogram, I believe that an approximating function method would be best. Actually, a three stage hybrid method using an approximation function in two stages and a spline fit in the other will be described.

In a typical one dimensional approximation problem we select a function $g(a_i,f)$ with which we wish to approximate the function $h'(f)$. The parameters, a_i, are selected to minimize the error as defined by some measure of error $E\{g(a_i,f), h'(f)\}$. In this problem we shall account for the time dependence by

using time dependent parameters, $a_i(t)$. The time dependence of $a_i(t)$ can be found using points from sets of ionograms. The choice of an approximating function for $g(a_i,f)$ is critical as regards the number of parameters, a_i, required for a suitably good approximation. A set of orthonormal functions is frequently selected as a set of basis vectors for analytical convenience in evaluating a_i. However, in our case we shall use a gradient method[5] for finding a_i so that we may select a $g(a_i,f)$ so as to minimize the number of a_i instead. To promote convergence of our gradient method of selecting a_i we shall use a least squares measure of error (ℓ_2 norm) which should be satisfactory from a philosophical standpoint.

In selecting the approximating function $g(a_i,f)$ we consider the general shape of an ionogram shape as shown in Fig. TR-1-2. We notice the following features:

1. $h'(f_1) = 0$

2. $\left|\frac{dh'}{df}\right|_{f=f_1} \longrightarrow \infty$

3. $h'(f_2) \rightarrow \infty$

A typical orthonormal function expansion would require many terms to approximate these features. Therefore we consider the following function,

$$g(a_i, f) = a_1(\ln f/f_1)^{a_2}, \quad 0 < a_2 < 1, \; f_1 < f < a_3$$

$$g(a_i, f) = a_4 + a_5(\ln f) + a_6(\ln f)^2, \quad a_3 < f < a_7$$

$$g(a_i, f) = a_8(\ln f_2/f)^{-a_9}, \quad a_7 < f < f_2$$

Notice that the points of the piecewise function are included in the parameter set. This function clearly satisfies all three of the features of $h'(f)$ listed above. If we constrain $g(a_i, f)$ and its first derivative to be continuous at the joints then we will have only five parameters to adjust for the least mean square error fit; i.e.,

$$g(a_i, a_3^-) = g(a_i, a_3^+)$$

$$\frac{\partial g}{\partial f}(a_i, a_3^-) = \frac{\partial g}{\partial f}(a_i, a_3^+)$$

$$g(a_i, a_7^-) = g(a_i, a_7^+)$$

$$\frac{\partial g}{\partial f}(a_i, a_7^-) = \frac{\partial g}{\partial f}(a_i, a_7^+)$$

and we find $\{a_i\}$ such as to minimize

$$[h'(f) - g(a_i,f)]^2$$

subject to the constraints on continuity. Using the four continuity constraints we eliminate four of the a_i's leaving five a_i's which we can renumber and assign as elements of a minimizing vector to be found using a gradient technique.

The steps involved in finding an approximation are:

1. Find a function, $g(a_{ij},f)$, approximating the j^{th} ionogram $h'_j(f,t)$.

2. Select a set of points, $\{g(a_{ij},f_m),f_m\}$ lying in the j^{th} observation and apply spline interpolation in time between successive ionograms to find a set of curves $\{g'(f_m,t)\}$. The intersection of these curves with a plane of constant time, $t = t_k$, gives a set of points, $\{g'(f_m,t_k)\}$. This set of points approximates ionogram data obtained at the same instant in time.

3. Find a function $g(a_i(t_k),f)$ giving the least mean square error fit to the point set $\{g'(f_m,t_k)\}$. This function $g(a_i(t_k),f)$ is our approximation to an

ionogram taken all at the same time instant, t_k.
From this we can select a set of points, $\{g(a_i(t_k), f_m)\}$, from which to compute $N(h, t_k)$.

In finding the approximating function in step 1 above we may use a weighted least squares cost function to discriminate between good and bad data points from the original ionogram data. We define the cost function for the j^{th} ionogram data as

$$J_j = \sum_{n=1}^{N} w_n \left[g(a_{ij}, f_n) - h'_j(f_n) \right]^2$$

where the weighting function, w_n, is made larger for well defined points, $h'(f_n)$, and smaller for poorly defined points. Assignment of w_n would be by operator judgement using some arbitrary standard scale.

Approximation of N(h,t)

The basic problem of determining N(h) from h'(f) data (or from $g(a_i(t_k),f)$ which approximates $h'(f,t_k)$) appears to be a problem in nonlinear estimation since we are given a set of noisy observations and some nonlinear mapping function relations N(h) and h'(f). If we were able to define a suitable approximating function for N(h), we should be able to find a least squares error fit to the observations using a gradient technique.

Examination of Fig. 3 shown in reference[1], shows that a suitable approximating function might be

$$h = b_1 + b_2 e^{-b_3 \ell n\ N}$$

$$= b_1 + b_2 N^{-b_3}$$

or

$$N = \left(\frac{b_2}{h - b_1}\right)^{1/b_3}$$

where b_1, b_2, and b_3 are parameters to be determined. These parameters can be written as elements of a state vector

$$Z = (b_1, b_2, b_3)$$

The above approximating function for N(h) can be used to find (using inverse mapping) $\overline{h'(f)}$, the h'(f) data computed from N(h). A cost functional is now defined as

$$J = \sum_{i=1}^{N} A_i (\overline{h'_z(f_n)} - h'_z(f_n))^2$$

$$+ \sum_{i=1}^{M} B_i (\overline{h'_o(f_n)} - h'_o(f_n))^2$$

$$+ \sum_{i=1}^{L} C_i (\overline{h'_x(f_n)} - h'_x(f_n))^2$$

where the subscripted z, o, and x refer to the three major types of waves producing the ionogram and the weighting factors A_i, B_i, and C_i would account for the relative quality of the data points.

To apply the time position correction described in the previous section we would replace $h'(f_n)$ by $g(a_i(t_k), f_n)$ in the expression for J. In this case the point weighting has already been accounted for in determining $g(a_i(t_k), f_n)$ so that $A_i = B_i = C_i$ in the expression for J.

A gradient technique [5] can now be used to determine the state vector

$$Z = (b_1, b_2, b_3)$$

such as to minimize the cost function, J. Therefore, we will have determined a least mean square error fit of our approximating function, N(h), to the observations, h'(f), or $N(h,t_k)$ using $g(a_i(t_k),f)$.

Conclusion

Methods have been described for finding N(h) or $N(h,t_k)$ from ionograms using approximation theory. A gradient technique is proposed to find the parameters of the approximating function. The success of the technique will depend upon the following factors:

1. The appropriate selection of the approximating model.

2. The speed of convergence of the gradient method computation of approximation model parameters.

The approximation model proposed for N(h) is quite simple and when used with h'(f) data would lead to results which could be compared directly with present N(h) results. Therefore, I recommend that this portion of the proposed method be evaluated first. Since the proposed model is so simple an additional term involving two additional parameters would not impose a computational hardship if experiments indicate a more accurate model is required.

Approximating $h'(f,t_k)$ by $g(a_i(t_k),f)$ appears to be a more difficult problem since estimation in two dimensions, frequency

and time, is required. Also the form of the approximating function appears to be more complex than for N(h). Another factor is that results obtained by accounting for the time/displacement changes over a single ionogram could not be compared directly with results obtained by current methods. For these reasons I recommend that the approximation of $h'(f,t_k)$ be attempted as a second step following the approximation of N(h).

BIBLIOGRAPHY

(1) J. E. Jackson, "The Reduction of Topside Ionograms to Electron Density Profiles", IEEE Proceedings, Vol. 57, No. 6 (June 1969) pp. 960-975.

(2) R. G. Madsen, "Isis Topside Sounder Data Processor", Study Report No. 93840, Astrodata, Inc.

(3) J. R. Rice, The Approximation of Functions, Vol. I, Addison Wesley, 1964.

(4) G. Birkhoff and H. L. Garabedian, "Smooth Surface Interpolation",

(5) J. F. Kinkel, "A Gradient Method with Improved Convergence", 9 September 1969.

(6) R. W. Hamming, Numerical Methods for Engineers and Scientists, McGraw-Hill, 1962.

Prepared for:

Astrodata, Inc.
240 E. Palais Rd.
Anaheim, Calif.

Astrodata Project 938400

TR-2

A WEIGHTED LEAST SQUARES APPROXIMATION METHOD

November 21, 1969

John F. Kinkel
Technical Consultant
Newport Beach, Calif.

A WEIGHTED LEAST SQUARES APPROXIMATION METHOD*

Given a function f(x) in terms of a set of data points $\{X_i, f(X_i)\}$, $i = 1,\ldots,N$, we wish to approximate f(x) by the function $F(X, \underline{\alpha})$, $\underline{\alpha} = (\alpha_1,\ldots,\alpha_K)^T$, where $\underline{\alpha}$ is a parameter set which we will select to minimize the weighted mean square error of the approximation. We approximate the differential of $F(X, \underline{\alpha})$ at X_i as

$$\Delta F(X_i, \underline{\alpha}) = \frac{\partial F(X_i, \underline{\alpha})}{\partial \alpha_1} \Delta\alpha_1 + \ldots. + \frac{\partial F(X_i, \underline{\alpha})}{\partial \alpha_K} \Delta\alpha_K$$

$$= \nabla_{\underline{\alpha}} F(X_i, \underline{\alpha})\ \Delta\underline{\alpha} \qquad i = 1, \ldots., N$$

$$\text{where } \nabla_{\underline{\alpha}} F(X_i, \underline{\alpha}) = \left(\frac{\partial F(X_i, \underline{\alpha})}{\partial \alpha_1}, \ldots., \frac{\partial F(X_i, \underline{\alpha})}{\partial \alpha_K}\right)$$

$$\Delta\underline{\alpha} = (\Delta\alpha_1, \ldots., \Delta\alpha_K)^T$$

We would like to find the ΔF such that

$$f(X_i) - F(X_i, \underline{\alpha}) - \Delta F(X_i, \underline{\alpha}) = 0, \quad i = 1, \ldots., N$$

or $$\Delta F(X_i, \underline{\alpha}) = f(X_i) - F(X_i, \underline{\alpha}), \qquad i = 1, \ldots., N$$

Now we wish to find the $\Delta\underline{\alpha}$ which will produce this result. We have

---

*A. J. Mallinckrodt, "A Weighted Least-Squares Adjustment for Network Synthesis" Astrodata Technical Report, July 1964.

$$\begin{bmatrix} f(X_1) - F(X_1, \underline{\alpha}) \\ \vdots \\ f(X_N) - F(X_N, \underline{\alpha}) \end{bmatrix} = \begin{bmatrix} \nabla_{\underline{\alpha}} F(X_1, \underline{\alpha}) \\ \vdots \\ \nabla_{\underline{\alpha}} F(X_N, \underline{\alpha}) \end{bmatrix} \Delta\underline{\alpha}$$

$$\underline{e} = A\ \Delta\underline{\alpha}$$

$$\underline{e} = \begin{bmatrix} f(X_1) - F(X_1, \underline{\alpha}) \\ \vdots \\ f(X_N) - F(X_N, \underline{\alpha}) \end{bmatrix}, \quad A = \begin{bmatrix} \nabla_{\underline{\alpha}} F(X_1, \underline{\alpha}) \\ \vdots \\ \nabla_{\underline{\alpha}} F(X_N, \underline{\alpha}) \end{bmatrix}$$

and the weighted least mean square estimate for $\Delta\underline{\alpha}$ is

$$\Delta\underline{\alpha}^* = (A^T Q A)^{-1}\ A^T Q \underline{e}$$

where Q is a symmetric positive definite N × N matrix. Now we iterate on $\underline{\alpha}$ producing sequentially

$$\underline{\alpha}^{m+1} = \underline{\alpha}^m + \Delta\underline{\alpha}^*$$

We continue iterating on $\underline{\alpha}$ until

$$\Delta\underline{\alpha}^* \longrightarrow 0$$

Now we inquire as to the error criteria imposed on $\underline{e}$ by this method. We note that

$$\Delta\underline{\alpha}^* = 0 \ <\ \ >\ A^T Q \underline{e} = \underline{0}$$

$$\Longleftrightarrow \frac{\partial F(X_1, \underline{\alpha})}{\partial \alpha_k} \sum_{j=1}^{N} Q_{1j} e_j + \ldots. + \frac{\partial F(X_N, \underline{\alpha})}{\partial \alpha_k} \sum_{j=1}^{N} Q_{Nj} e_j = 0$$

$$k = 1, \ldots., K$$

Now consider the cost functional defined by

$$J = \underline{e}^T Q \underline{e} = \sum_{i=1}^{N} \sum_{j=1}^{N} e_i Q_{ij} e_j$$

$$= e_1 \sum_{j=1}^{N} Q_{1j} e_j + \ldots. + e_N \sum_{j=1}^{N} Q_{Nj} e_j$$

$$\min_{\underline{\alpha}} J \quad <\!\!> \quad (\nabla_{\underline{\alpha}} J)^T = 0 \Longleftrightarrow \frac{\partial J}{\partial \alpha_k} = 0, \quad k = 1, \ldots., K$$

$$\frac{\partial J}{\partial \alpha_k} = \sum_{i=1}^{N} \sum_{j=1}^{N} \left(\frac{\partial e_i}{\partial \alpha_k} Q_{ij} e_j + e_i Q_{ij} \frac{\partial e_j}{\partial \alpha_k} \right)$$

$$= 2 \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{\partial e_i}{\partial \alpha_k} Q_{ij} e_j \quad \text{since } Q_{ij} = Q_{ji}$$

But $\dfrac{\partial e_i}{\partial \alpha_k} = - \dfrac{\partial F(X_i, \underline{\alpha})}{\partial \alpha_k}$

$$\therefore \frac{\partial J}{\partial \alpha_k} = -2 \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{\partial F(X_i, \underline{\alpha})}{\partial \alpha_k} Q_{ij} e_j, \quad k = 1, \ldots., K$$

We see that

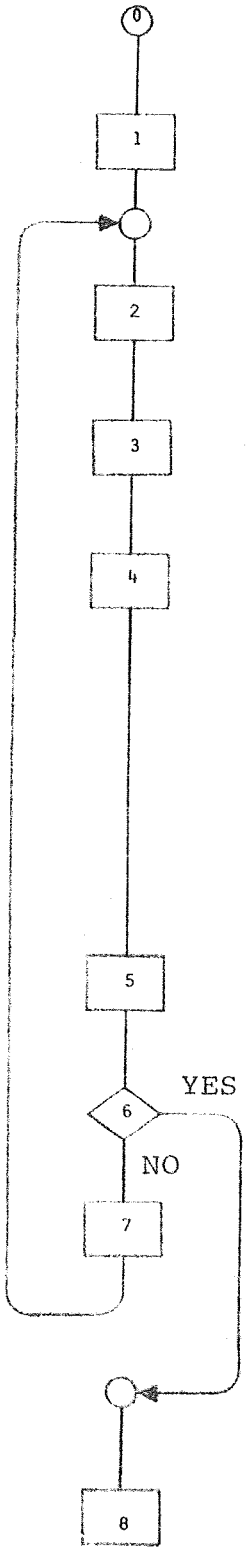
$$(\nabla_{\underline{\alpha}} J)^T = \underline{0} \iff \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{\partial F(X_i, \underline{\alpha})}{\partial \alpha_k} Q_{ij} e_j = 0, \quad k = 1, \ldots, K$$

and since $\partial \underline{\alpha}^* = 0 \iff \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{\partial F(X_i, \underline{\alpha})}{\partial \alpha} Q_{ij} e_j = 0$

we have $\Delta \underline{\alpha}^* = 0 \implies (\nabla_{\underline{\alpha}} J)^T = \underline{0} \implies \min_{\underline{\alpha}} J$

Therefore, by iterating on $\underline{\alpha}$ until we have achieved $\Delta \underline{\alpha}^* = 0$ we have actually achieved a weighted least squares fit to f(x) by $F(X, \underline{a})$. The desired result is achieved by sneaking in through the back door.

We see also that if J were quadratic in $\underline{\alpha}$ then F would be linear in $\underline{\alpha}$ so that $\Delta F = \nabla_{\underline{\alpha}} F \cdot \Delta \underline{\alpha}$ would be exactly true and convergence to $\Delta \underline{\alpha} = 0$ would be achieved in exactly one step as with the Newton-Raphson method. For the case in which F is linear in $\underline{\alpha}$, J quadratic in $\underline{\alpha}$, the Mallinckrodt* and Newton-Raphson methods are identical. An advantage of the Mallinckrodt algorithm over the Newton-Raphson is that only first order partial derivative are required while second order partials are required for the Newton-Raphson. See Fig. TR-2-1 for a flow diagram for the least squares determination of N(h).



START:

1. Initialize α_j, $j = 1,K$

2. Compute h_i, $i = 1,N$, $\ni$ $\bar{f}_i = f_i$ †

3. Compute $\bar{h}_i$, $i = 1,N$ (Madsen Inverse Mapping)

4. Do 20 $j = 1,K$
   Perturb α_j
   Do 10 $i = 1,N$
   Compute h^*_{ij} $\ni$ $\bar{f}_i^* = f_i$ †
   Compute $\bar{h}^*_{ij}$ (Madsen Inverse
   10 Compute $A_{ij} = \bar{h}^*_{ij} - h$
   20 Restore α_j

   } Compute A_{ij}

5. Compute $\Delta\underline{\alpha} = (A^T Q A)^{-1} A^T Q(\underline{h}' - \underline{\bar{h}}')$

6. $|\Delta\underline{\alpha}| < \varepsilon$?

7. $\underline{\alpha} = \underline{\alpha} + \Delta\underline{\alpha}$

8. $N = F(h, \underline{\alpha})$

Fig. TR-2-1. Flow Diagram - Least Squares Determination of N(h)

† See Technical Report 3

Prepared for:

Astrodata, Inc.
240 E. Palais Rd.
Anaheim, Calif.

Astrodata Project 938400

TR-3

INVERSE MAPPING TO SPECIFIED f

November 24, 1969

John F. Kinkel
Technical Consultant
Newport Beach, Calif.

INVERSE MAPPING TO SPECIFIED f

We would like to perform the inverse mapping operation such that the frequency, f, of the end point is prespecified, as shown in Fig. TR-3-1.

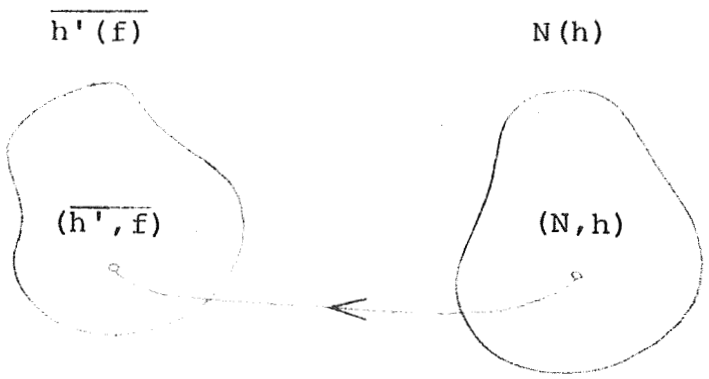


Fig. TR-3-1

To do this we have the following relations for the X-Trace:

$$\overline{f}_x = \frac{1}{2} f_H + \frac{1}{2} \sqrt{f_H^2 + 4f_N^2} \tag{1}$$

Subtracting 1/2 f_H from both sides and squaring gives

$$\overline{f}_x^2 - f_H \overline{f}_x - f_N^2 = 0 \tag{2}$$

According to Jackson [13]

$$f_H = 2.8\ B \tag{3}$$

$$f_N = \sqrt{\frac{N}{12400}} \tag{4}$$

From our approximation formula for N we have

$$N = F(h, \underline{\alpha}) \tag{5}$$

A representative curve of this function is shown in Fig. TR-3-2.

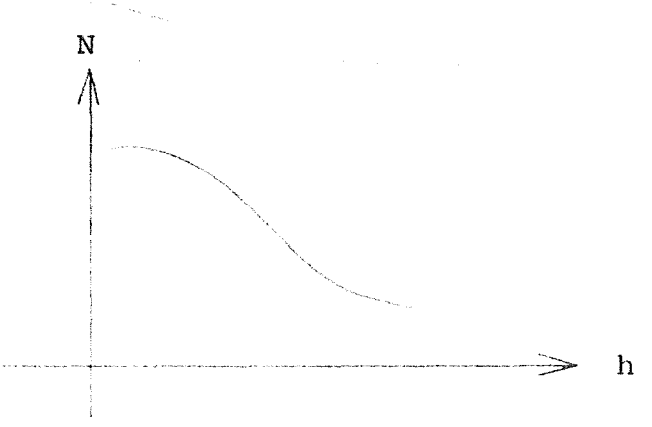


Fig. TR-3-2

We see that for any specified value of h we may compute

N from eqn. (5)

f_N from eqn. (4)

B from FIELDG[17] and interpolation

f_H from eqn. (3)

$\bar{f}$ from eqn. (1)

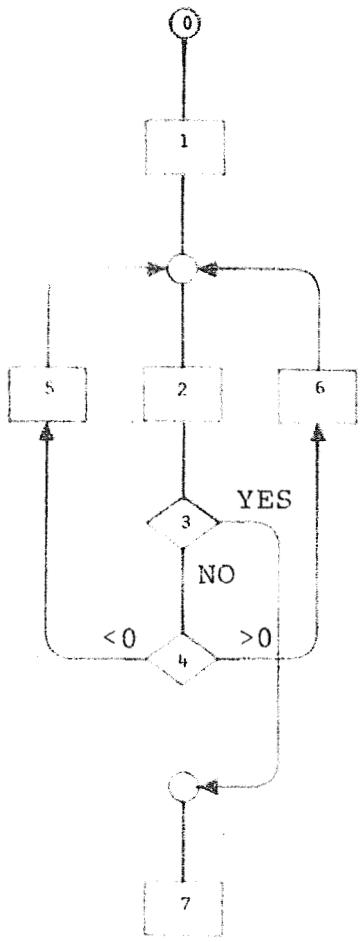
Since N and B are monotone decreasing in h, f_N and f_H are also monotone decreasing. Therefore we conclude that $\bar{f}$ is likewise monotone decreasing in h. This property permits us to use a

successive approximation technique for finding h $\ni \left|\frac{\bar{f} - f}{f}\right| \to 0$.

A convenient algorithm for this purpose is Bolzano's root finding technique.*

We may elect to select h to within some given tolerance or to iterate until $0 \leq \left|\frac{\bar{f} - f}{f}\right| < \varepsilon$. Figures TR-3-3, TR-3-4, and TR-3-5 show h selected to within .01% of a maximum value.

*Wilde, Douglas J., Optimum Seeking Methods, Prentice Hall, 1964.



START:

1. Initialize parameters
   $h = h_{MAX}/2$, $\Delta = h/2$

2. Compute N, f_N, B, f_H, $\bar{f}_x$, $\Delta = \Delta/2$

3. $\dfrac{|\bar{f}_x - f_x|}{f_x} < \varepsilon$?

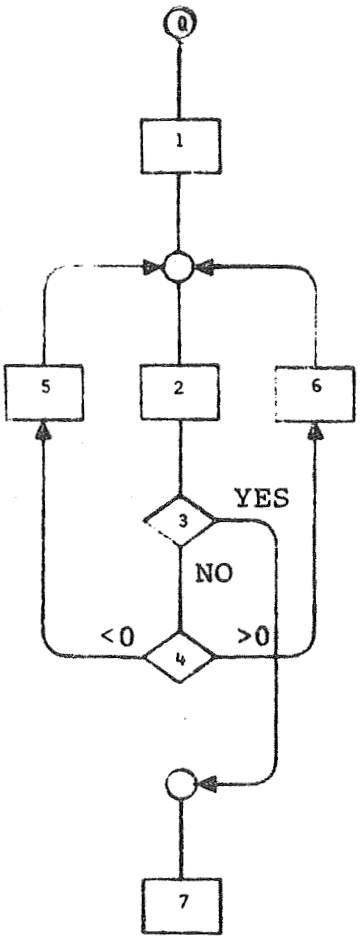
4. $\bar{f}_x - f_x$

5. $h = h - \Delta$

6. $h = h + \Delta$

7. $N = F(h, \underline{\alpha})$
   $f_N = \sqrt{N/12400}$
   B = from existing subroutine
   $f_H = 2.8\ B$
   $\bar{f}_x = \frac{1}{2} f_H + \frac{1}{2} \sqrt{f_H^2 + 4f_N^2}$

Fig. TR-3-3. Flow Diagram - Selection of h ∋ (N,h) ⟶ $(\bar{h}; \bar{f}_x)$ Where $\bar{f}_x$ is Specified



START:

1. Initialize parameters
   $h = h_{MAX}/2$, $\Delta = h/2$

2. Compute N, f_N, $\overline{f_o} = f_N$, $\Delta = \Delta/2$

3. $\dfrac{|\overline{f}_o - f_o|}{f_o} < \varepsilon$?

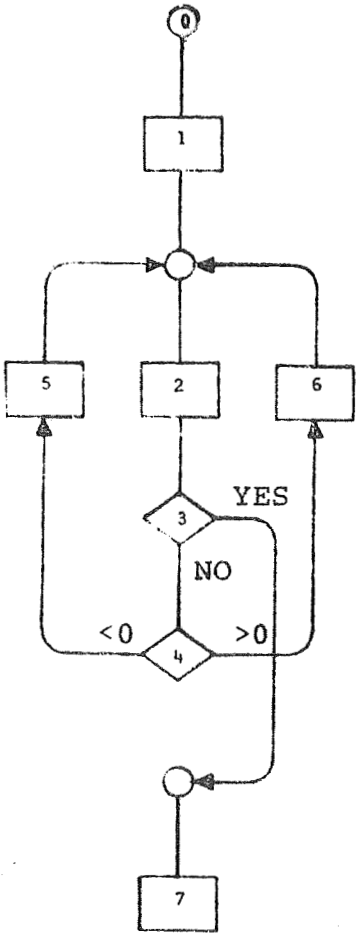
4. $\overline{f}_o - f_o$

5. $h = h - \Delta$

6. $h = h + \Delta$

7. $N = F(h, \underline{\alpha})$
   $f_N = \sqrt{N/12400}$

Fig. TR-3-4. Flow Diagram - Selection of h $\ni$ $(N,h) \longrightarrow (\overline{h};\overline{f}_o)$ Where $\overline{f}_o$ is Specified



START:

1. Initialize parameters
   $h = h_{MAX}/2$, $\Delta = h/2$

2. Compute N, f_N, B, f_H, $\overline{f}_z$, $\Delta = \Delta/2$

3. $\dfrac{\left|\overline{f}_z - f_z\right|}{f_z} < \varepsilon$?

4. $\overline{f}_z - f_z$

5. $h = h - \Delta$

6. $h = h + \Delta$

7. $N = F(h, \underline{\alpha})$
   $f_N = \sqrt{N/12400}$
   B = from existing subroutine
   $f_H = 2.8\ B$
   $\overline{f}_z = \frac{1}{2} f_H + \frac{1}{2} \sqrt{f_H^2 + 4 f_N^2}$

Fig. TR-3-5. Flow Diagram - Selection of $h \ni (N,h) \longrightarrow (\overline{h}; \overline{f}_z)$ Where $\overline{f}_z$ is Specified

Prepared for:

Astrodata, Inc.
240 E. Palais Rd.
Anaheim, Calif.

Astrodata Project 938400

TR-4

SIMPLIFICATION OF MATRIX INVERSION PROBLEM

December 1, 1969

John F. Kinkel
Technical Consultant
Newport Beach, Calif.

SIMPLIFICATION OF MATRIX INVERSION PROBLEM

We are given the problem

$$AX = b$$

from which we wish to find

$$X = A^{-1} b$$

A Gauss-Jordan inversion process starts with

$$(A \vdots I \vdots b)$$

and through a series of row operations becomes

$$(I \vdots A^{-1} \vdots X)$$

Ordinarily the last column of the augmented matrix is omitted and X is found by a subsequent multiplication.

$$X = A^{-1} b$$

For the special case in which A is a symmetric matrix we may simplify the procedure to reduce only the upper right triangle of the left matrix to 0's which will produce only the upper right triangle and diagonal of A^{-1}. We know that if $A = A^T$,

then $(A^{-1})^T = A^{-1}$ so that the lower left triangle of A^{-1} is found from

$$(A^{-1})ij = (A^{-1})ji$$

This would reduce the number of operations from $N^2(N + 1)$ for a savings of $2N^2 - 3N$.

$$\left(\sum_{i=1}^{N} i = \frac{1}{2} N(N + 1) \right)$$

However, when A^{-1} is not required for some other purpose than finding X, an even greater economy is available by deleting the I matrix augmentation. A further advantage is that A need not be symmetric. Here we would have

$$(A \vdots b) \rightarrow (I \vdots X)$$

Not only would we save $\frac{N - 1}{2N} N^2(N + 1) = \frac{1}{2} N(N^2 - 1)$ operations in the inversion process but also another N^2 operations in the multiplication, $X = A^{-1}b$, which has been eliminated. Therefore, the total savings in this method amounts to $\left\{\frac{1}{2} N(N^2 - 1) + N^2\right\}$ as compared with the standard inversion and subsequent multiplication.

For example, consider a 10 × 10 matrix; i.e., N = 10. The number of operations in finding the inverse is

$$N^2(N + 1) = 1100$$

The subsequent multiplication requires an additional $N^2 = 100$ for a total of 1200 operations.

Finding X directly and not A^{-1} requires $\frac{1}{2} N(N + 1)^2 = 605$ operations for a savings of 595.

This agrees with the expression for savings of $\left\{\frac{1}{2} N(N^2 - 1) + N^2\right\} = \frac{1}{2} 990 + 100 = 595$. Therefore, for N reasonably large, say $N \geq 10$, the savings amounts to almost 50%.

Subroutine MXV for computing $\underline{x}$, where

$$A\underline{x} = \underline{b}$$

is listed on the following page.

MXV
list

Prepared for:

Astrodata, Inc.
240 E. Palais Rd.
Anaheim, Calif.

Astrodata Project 938400

TR-5

TIME SKEW PROBLEM

December 12, 1969

John F. Kinkel
Technical Consultant
Newport Beach, Calif.

TIME SKEW PROBLEM

Fig. TR-5-1 shows the basic assumption made in the Jackson method of ionogram reduction. Perhaps instead of assuming successive values of N_i one could interpolate between successive ionograms. Since N_0 (electron density at satellite height) is the only value obtained without assumption of previous values, this would apply only for essentially constant satellite height between adjacent ionograms. Also, while interpolation would be possible in reducing the first ionogram, extrapolation would be required for the second. Interpolated and extrapolated values would be used in place of the assumed values of N_i shown in Fig. TR-5-1.

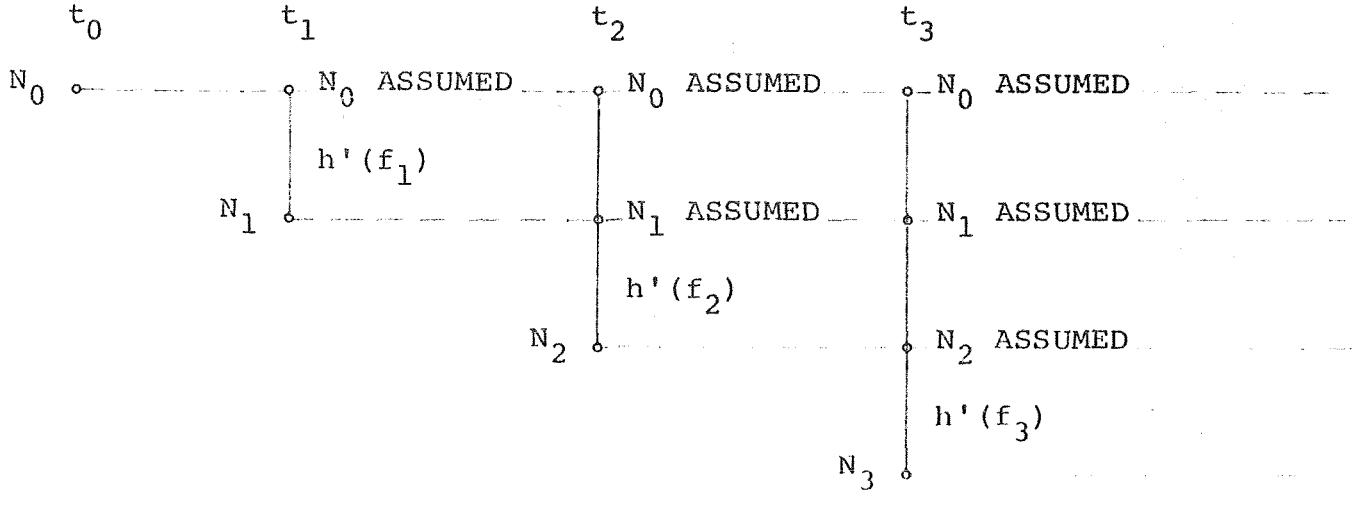


Fig. TR-5-1

For the least squares estimation approach an entirely equivalent set of assumptions is made as shown in Fig. TR-5-2.

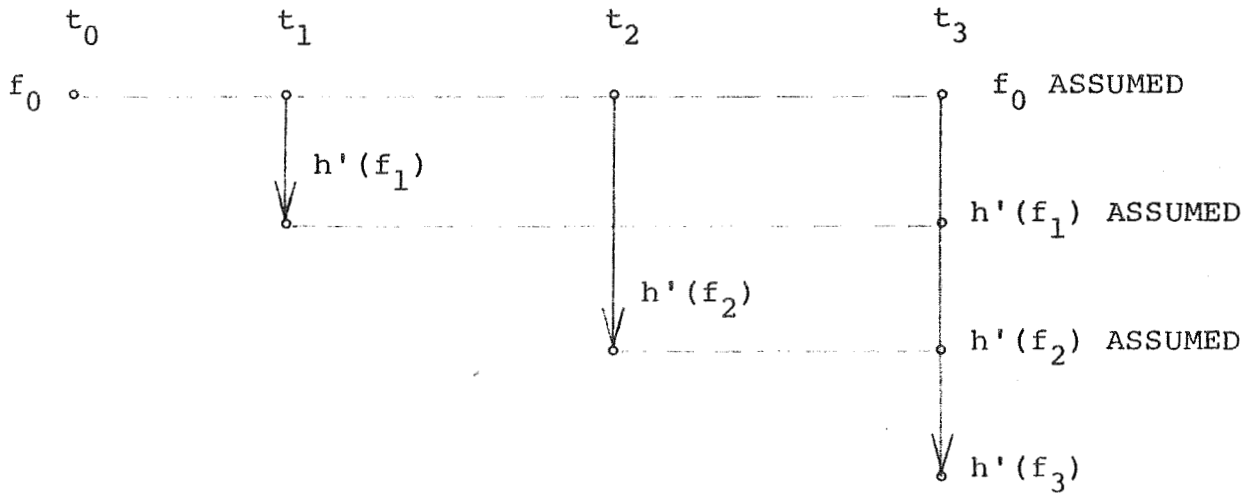


Fig. TR-5-2

Only four data points are shown for convenience.

The concept of interpolating $h'(f_i)$ data points from successive ionograms is immediately obvious from the last figure. Unlike the case for N_i values, only interpolation (no extrapolation) is required for $h'(f_i)$ values.

Since the approach of interpolating h'(f) data is regarded with suspicion by knowledgeable people skilled in the art, and since solution by interpolation of N(h) is fraught with difficulty, the following approach is proposed.

1. Interpolate h'(f) as described above.

2. Compute N(h) from interpolated h'(f) data.

3. Interpolate N(h) as obtained in step 2 in order to compute $\overline{h'(f)}$.

4. If $\overline{h'(f)}$ agrees with h'(f), then we have found N(h) profiles which when interpolated yield results matching the physical measurements.

Prepared for:

Astrodata, Inc.
240 E. Palais Rd.
Anaheim, Calif.

Astrodata Project 938400

TR-6

OPTIMAL STEP SIZE BY ONE DIMENSIONAL SEARCH

December 30, 1969

John F. Kinkel
Technical Consultant
Newport Beach, Calif.

OPTIMAL STEP SIZE

BY ONE DIMENSIONAL SEARCH

The computation of step size using the Hessian matrix is based on the assumption that the cost functional can be reasonably well approximated by a truncated Taylor series having terms of no higher order than second. This assumption is frequently unjustified resulting in a step size which may be either too large or too small. For the case in which the step size is too small the cost reduction at each iteration is less than could be achieved resulting in the need for more iterations than necessary for convergence. A more serious defect occurs for the case in which the step size is too large; in this case the cost may increase for some iterations, seriously impairing, if not completely preventing, convergence to the optimal solution. This effect has been noted in practice.

Since we know that there exists some step size in the gradient direction which if sufficiently small will produce a decrease in cost, we should be able to produce a cost reduction in every iteration until the optimal solution is achieved. What we would like to do is to find the step size that will yield the greatest reduction in cost for each iteration. This optimal step size

may be found directly (as opposed to indirectly as before) by means of a one dimensional search technique. Before we can apply one dimensional search over an interval we need to define the interval, S*, to be searched, as shown in Fig. TR-6-1.

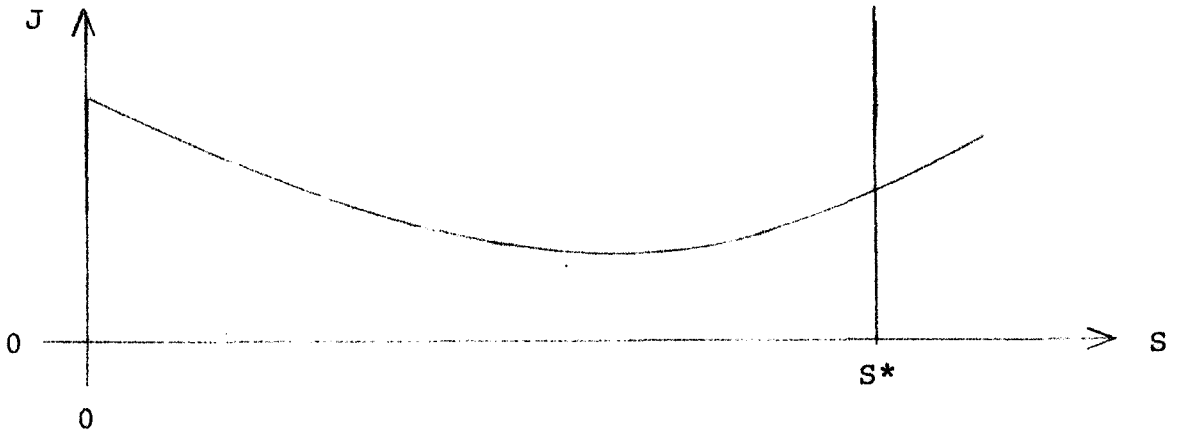


Fig. TR-6-1

For the one dimensional search technique to be effective, J(S), $0 \leq S \leq S^*$ must be unimodal. Therefore, we would like to define S* as

$$\min_{S} \frac{\partial J}{\partial S} > 0 \Longrightarrow S^*$$

Since we know there exists $\delta > 0$ such that $J(\delta) < J(0)$, it is clear that defining S* as shown guarantees that J(S) will be unimodal oerr the interval $0 < S < S^*$.

Having defined the interval, we can now apply either a Fibonacci or Golden Section search method.*

*Wilde, Douglas J., Optimum Seeking Methods, Prentice Hall, 1964

We can approximate S* by the following algorithm:

$$J(S_i) \triangleq J(\underline{z} - S_i \nabla_{\underline{z}} J) \qquad , S_0 = 0$$

$$J(S_i) < J(S_{i-1}) \Longrightarrow S^* > S_i, \; S_1 > 0$$

If $J(S_i) < J(S_{i-1})$, then let $S_{i+1} = KS_i$, $K > 1$, and repeat test. Continue this process until $J(S_i) \geq J(S_{i-1})$. Then let $S^* = S_i$. Perhaps $S_i = 10^{-3}$ and $K = 2$ would be good choices initially. Having now determined a value for S* we may search this interval for the value of $0 < S < S^*$ which minimizes J.

A search by Golden Section will be almost as efficient as a Fibonacci search and should be less subject to round-off error in the final steps. Twelve iterations will define S to within 0.5% of S*.

We can compare the burden of computation for the Golden Section search relative to the Hessian matrix method as follows. The cost must be computed once per iteration for the search method and once for each element of the Hessian matrix. The number of elements, K, to be computed in the Hessian matrix is given by

$$K = \frac{1}{2}(N + 1)N$$

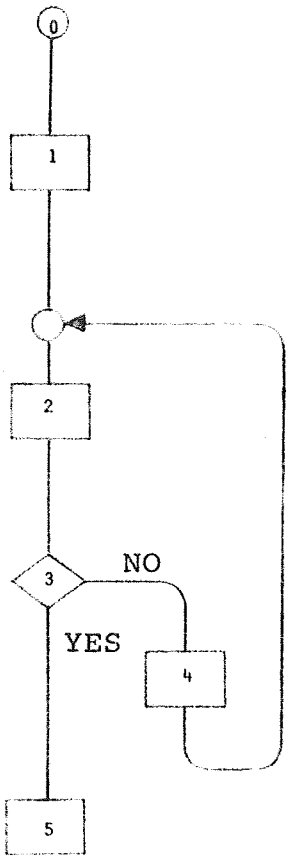
where N is the number of state variables.

N	K
2	3
3	6
4	10
5	15

If we assume that twelve iterations of the Golden Section search is adequate, then we see that for four or less state variables, the evaluation of the Hessian requires fewer computations but more computation for five or more state variables.

Refer to Fig. TR-6-2 and Fig. TR-6-3 for flow diagrams for the interval computation and the Golden Section Search for optimal step size.

Fig. TR-6-4 is used in conjunction with Fig. TR-6-1 to show the sequence of computing values in successive iterations in the Golden Section Search.



START:

Z_i, D_i

1. $Y_i = Z_i$, $i = 1, \ldots, N$
   $S = 10^{-3}$
   $E2 = E1$

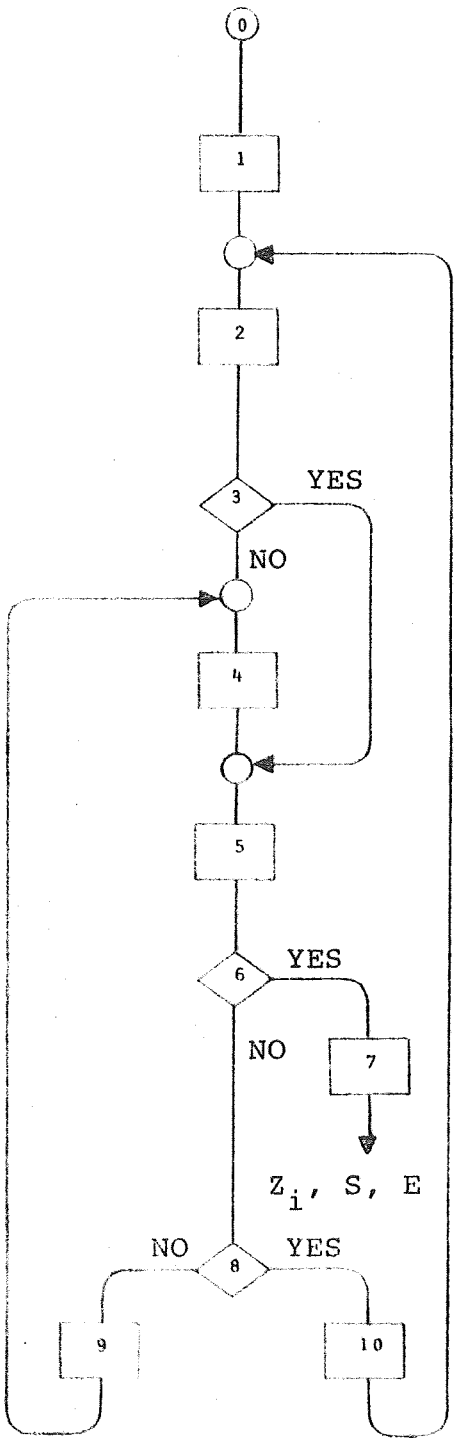
2. $Z_i = Y_i - SD_i$, $i = 1, \ldots, N$
   Compute E

3. $E \geq E2$?

4. $S = 2S$
   $E2 = E$

5. $S \ni J(S_i) \geq J(S_{i-1}) \Longrightarrow \frac{\partial J}{\partial S} > 0$

Fig. TR-6-2. Flow Diagram - Interval Computation



START:

Y_1, S

1. S1 = 0
   M1 = 0

2. $Z_i = Y_i - (0.382S + S1)D_i$, i = 1, ..., N
   Compute E
   E1 = E

3. M1 > 0?

4. $Z_i = Y_i - (0.618S + S1)D_i$, i = 1, ..., N
   Compute E
   E2 = E

5. M1 = M1 + 1

6. M1 = 12?

7. S = S + 1

8. E2 > E1?

9. E1 = E2
   S1 = S1 + .382S
   S = .618S

10. E2 = E1
    S = .618S

Fig. TR-6-3. Flow Diagram - Golden Section Search For Optimal Step Size

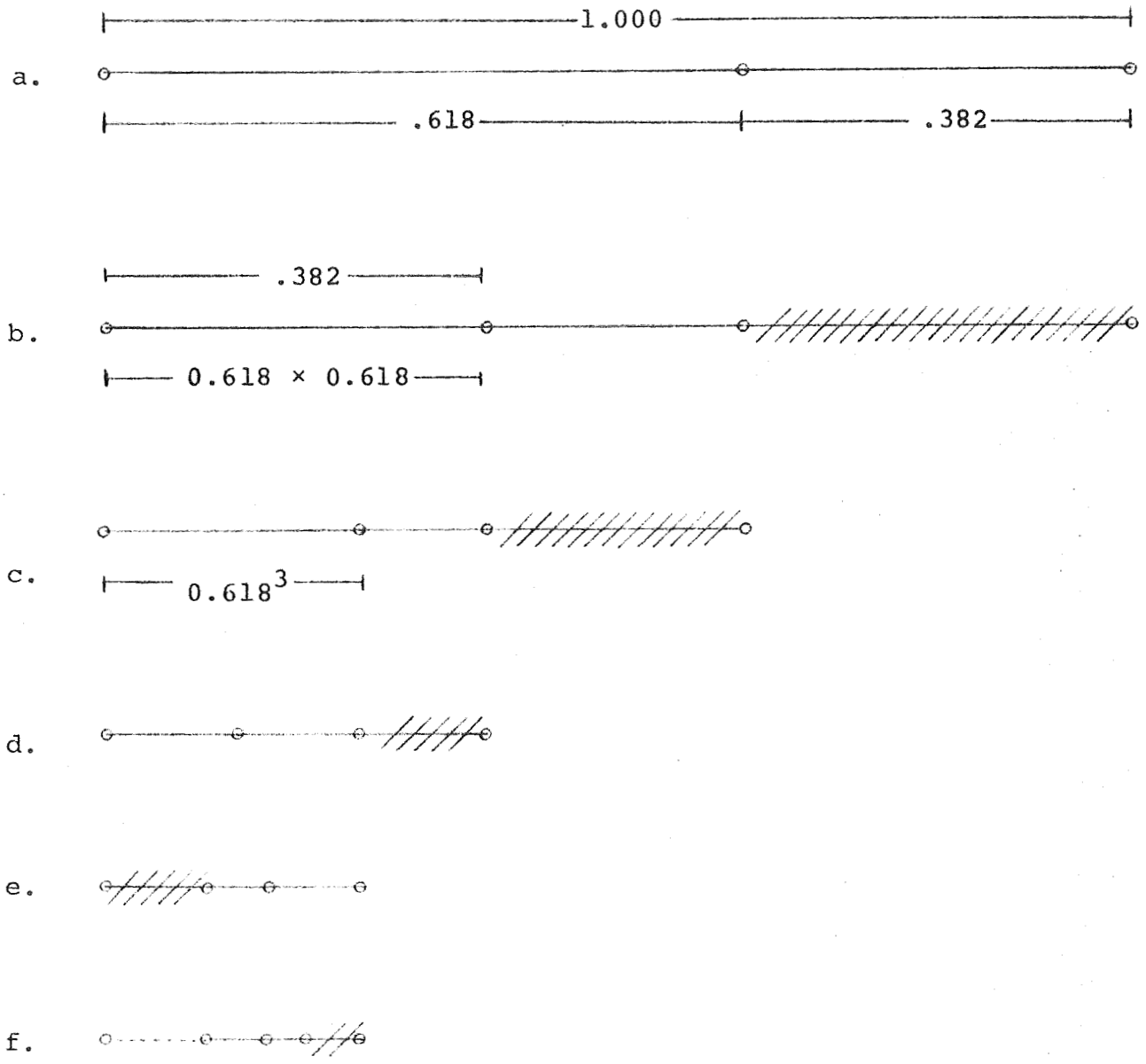


Fig. TR-6-4. Golden Section Search Value Computation Sequence

Prepared for:

Astrodata, Inc.
240 E. Palais Rd.
Anaheim, Calif.

Astrodata Project 938400

TR-7

HORIZONTAL PROCESSING

January 30, 1970

John F. Kinkel
Technical Consultant
Newport Beach, Calif.

HORIZONTAL PROCESSING

Introduction

Because of the motion of the satellite relative to the earth, it is a basic property of the swept frequency technique that within the same ionogram successive h'(f) data points correspond to soundings taken through the ionosphere at different earth based coordinates (see Fig. TR-7-1).

It is implicitly assumed in currently applied data processing methods for converting h'(f) data to N(h) profiles that the electron density profile is constant over the section of ionosphere traversed by the satellite during the period of one ionogram. This assumption is required not only for the lamination technique of Jackson[13] (forward processing) and Madsen* (inverse processing) but also for the least squares estimation technique described elsewhere in this report. The error in the electron density profile resulting from this assumption is not known.

Horizontal processing is a method of data reduction which does not require the assumption of constant N(h) profile over the section required by one ionogram. In horizontal processing electron density values along the satellite track are estimated

---

*See Section III, this report

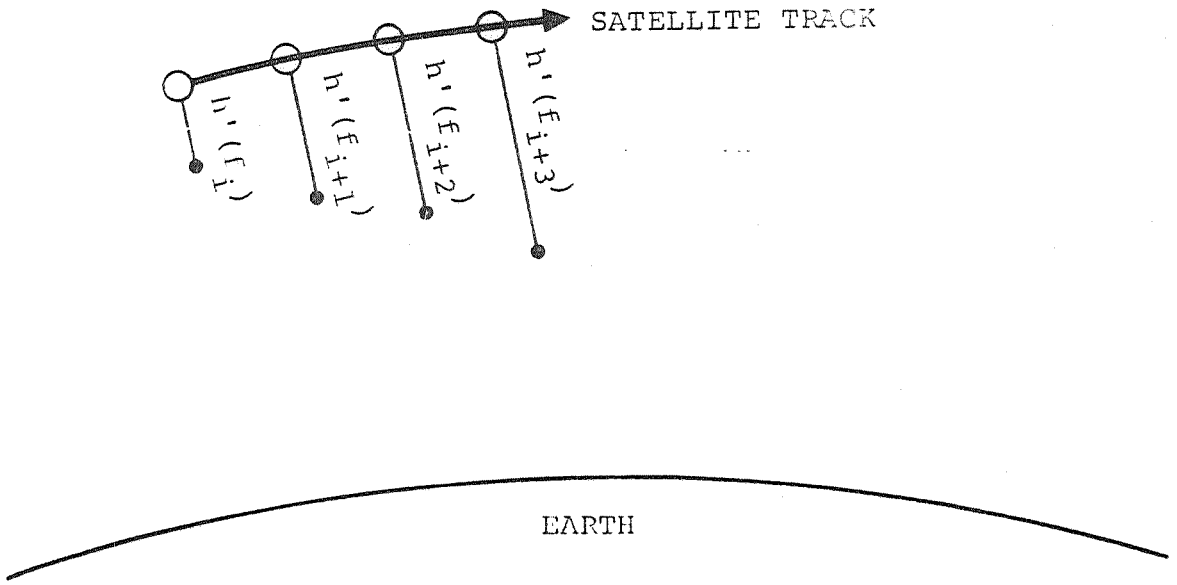


FIGURE 1

by linear interpolation between successive ionograms. Horizontal processing may be applied in conjunction with either lamination or least squares methods of data processing.

Lamination Method

In the lamination method it is essential to have a data point for the exit frequency so that the electron density at satellite height may be obtained. Although we would prefer to linearly interpolate electron density at constant height, since the satellite height cannot change greatly between successive ionograms, we may linearly interpolate to obtain the electron density for times between successive ionogram exit frequency data points. It is the interpolated electron density value that is used in the computation of electron density for each successive lamination. In the Jackson method of Forward Processing it is not clear how to select h'(f) points so that the lamination altitude will be the same for successive ionograms. With the Madsen method of Inverse Processing the lamination boundaries may be selected to be the same. All interpolations of electron density following the first at satellite height may be made at constant height. Of course, extrapolation rather than interpolation will be required for the last ionogram in a sequence.

The interpolation is illustrated in Fig. TR-7-2. At time t_{11} the point 1 is obtained from the exit frequency for ionogram 1. A

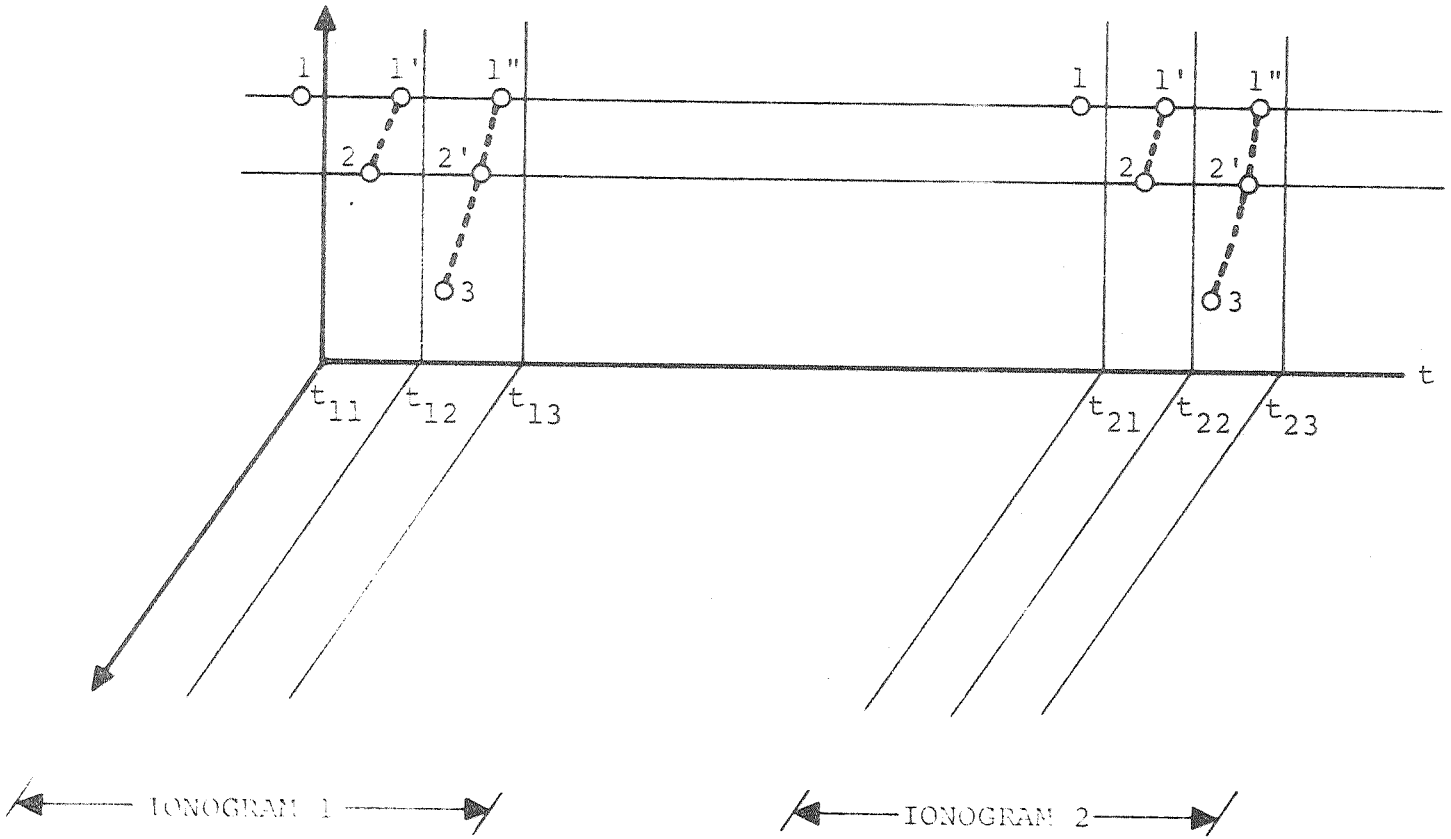


FIGURE 2

similar point is obtained for time t_{21} from ionogram 2. An h'(f) data point is selected corresponding to a time t_{12}. Point 1 is now extrapolated forward in time to t_{12}, giving point 1'. The electron density at point 1' is used to compute point 2. This process is continued until the last useful data point has been processed, say point j. Then we will have a complete N(h) profile corresponding to time t_{1j} for the first ionogram, t_{2j} for the second, etc.

In the lamination method there are two possible approaches for the interpolation of electron density at satellite height.

The first of these is just to find

$$N_j\left(h_{ij} + \frac{t_{2j} - t_{1j}}{t_{1,j+1} - t_{1j}}\,(h_{1,j+1} - h_{1j}),\ t_{2j}\right) = N_j(h_{1j},t_{1j})$$

$$+ \frac{t_{2j} - t_{1j}}{t_{1,j+1} - t_{1j}}\left[N_{j+1}(h_{1',j+1},t_{1,j+1}) - N_j(h_{1j},t_{1j})\right]$$

where

$N_j(h,t)$ = electron density at height h and time t from j^{th} ionogram

h_{ij} = height from i^{th} point from j^{th} ionogram

t_{ij} = time of i^{th} point from j^{th} ionogram

The second approach is to find the electron density from the ionogram having the higher satellite position at the satellite height of the adjacent ionogram without interpolation. This step is easily justified because of the very small time interval involved in this case. To be specific, suppose that

$$h_{1,j+1} > h_{1j}$$

Now, using the Madsen Inverse Processing technique, we can find

$$N_{j+1}(h_{1',j+1}, t_{2,j+1}) = N_{j+1}(h_{1j}, t_{2,j+1})$$

where we have selected $h_{1',j+1} = h_{1j}$. Using this value of N_{j+1} we can interpolate N_j at constant height as

$$N_j(h_{1j}, t_{2j}) = N_j(h_{1j}, t_{1j}) + \frac{t_{2j} - t_{1j}}{t_{1',j+1} - t_{1j}} \left[N_{j+1}(h_{1j}, t_{1,j+1}) - N_j(h_{1j}, t_{1j}) \right]$$

For either of these cases succeeding interpolations may be made at constant height as follows:

$$N_j(h_{ij}, t_{kj}) = N_j(h_{ij}, t_{ij}) + \frac{t_{kj} - t_{ij}}{t_{i,j+1} - t_{ij}} \left[N_{j+1}(h_{ij}, t_{i,j+1}) - N_j(h_{ij}, t_{ij}) \right]$$

By this means we may compute successive laminations as shown by Fig. TR-7-3.

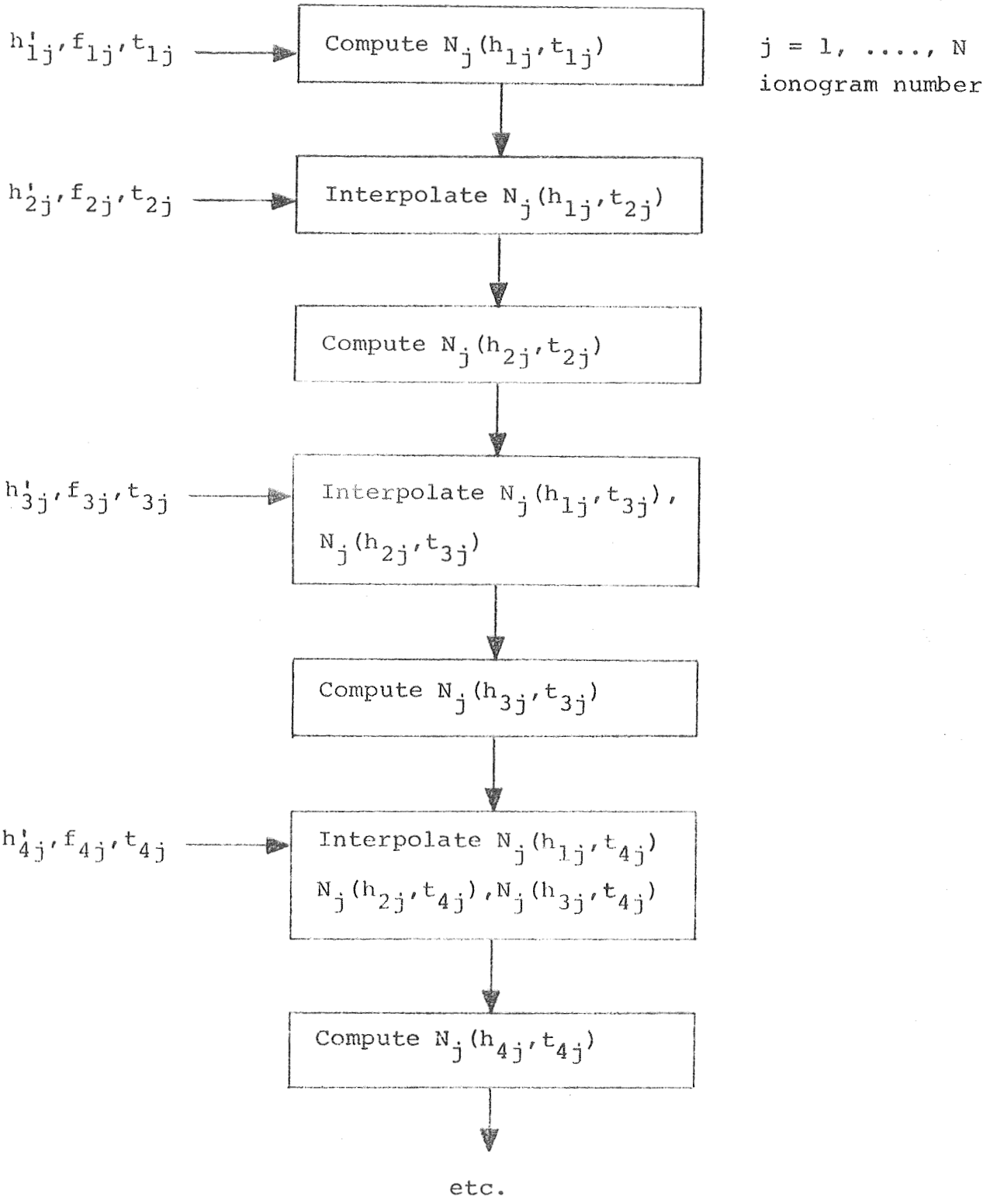


Fig. TR-7-3. Horizontal Processing Using Lamination Method

Difficulty will be encountered, of course, with ionogram sequences in which few successive complete ionograms are available. In these cases interpolation in conjunction with a least squares estimation may offer a solution.

Least Squares Estimation

For this method the initial estimate for N(h) is determined without extrapolation. Successive revisions to this estimate using interpolation are then determined iteratively in order to improve the estimate. As shown in Fig. TR-7-4, the initial time skewed N(h) profiles are determined without regard to change in N(h) profiles with position along the track. Using constant h (or nearly so) interpolation, a sequence of N(h) profiles corresponding to constant time are found. This sequence of constant time N(h) profiles is used in the inverse mapping process in the second iteration to find an improved estimate of the time skewed N(h) profile. All ionograms in a sequence are processed in this manner before proceeding to the next iteration. This process of estimating time skewed profiles followed by interpolation may be performed sequentially to improve the estimate. As a final step the time skewed N(h) profiles are interpolated to the desired constant time N(h) profile.

We know that in actuality the electron density profile is a function of position along the satellite trace so that we may

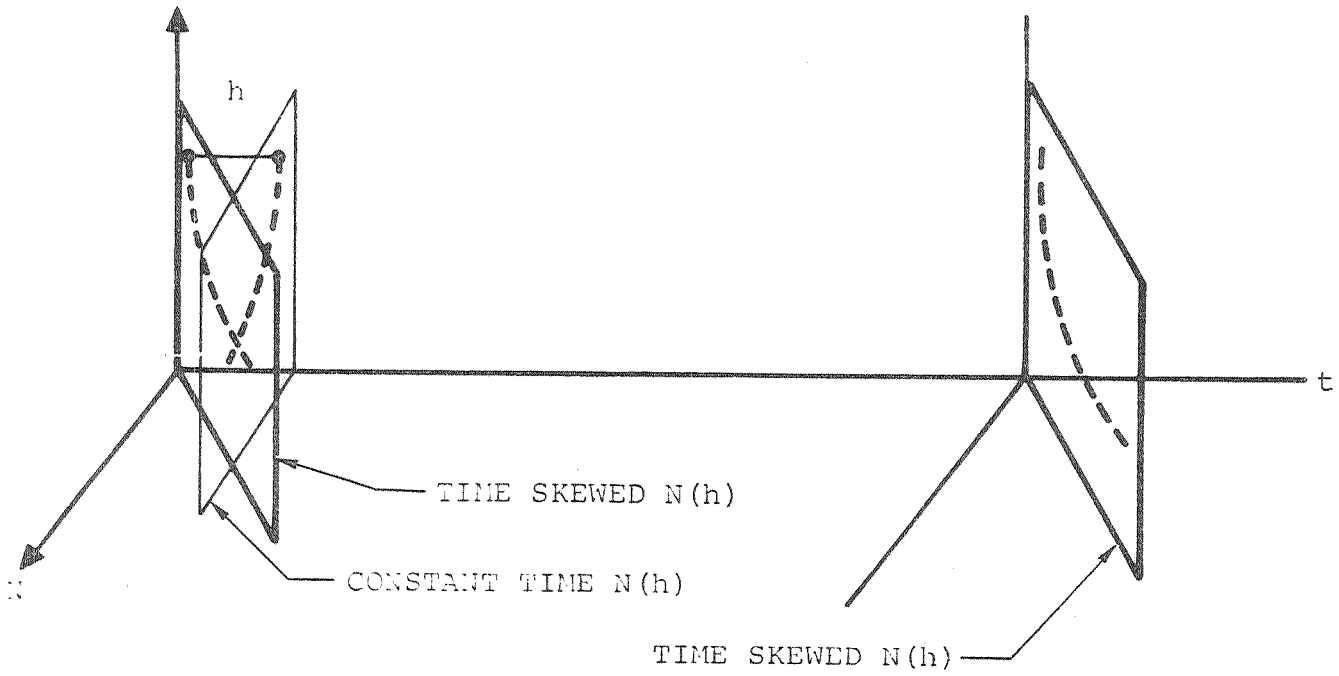


FIGURE 3

write the electron density approximation

$$N(h, \underline{\alpha}, t)$$

For the least squares method we need to use the inverse mapping

$$\left\{N_i(h_i, \underline{\alpha}, t_j),\ i = 1, \ldots, j\right\} \longrightarrow \bar{h}'_j(f_j),\ j = 1, \ldots, J$$

where J = number of h'(f) data points.

Since t and f are uniquely related within the time period of one ionogram we may write that

$$f_j \leftrightarrow t_j$$

For the first iteration we merely use

$$\left\{N_i(h_i, \underline{\alpha}),\ i = 1, \ldots, j\right\} \longrightarrow \bar{h}'_j(f_j)$$

However, for subsequent iterations we find

$N_i(h_i, \underline{\alpha}, t_j)$ projected on constant time plane

by linear interpolation as described in the preceding section. Therefore all subsequent iterations make use of

$$\left\{N_i(h_i, \underline{\alpha}, t_j),\ i = 1, \ldots, j\right\} \longrightarrow \bar{h}'_j(f_j)$$

For the least squares method this process is shown in the flow diagram of Fig. TR-7-5.

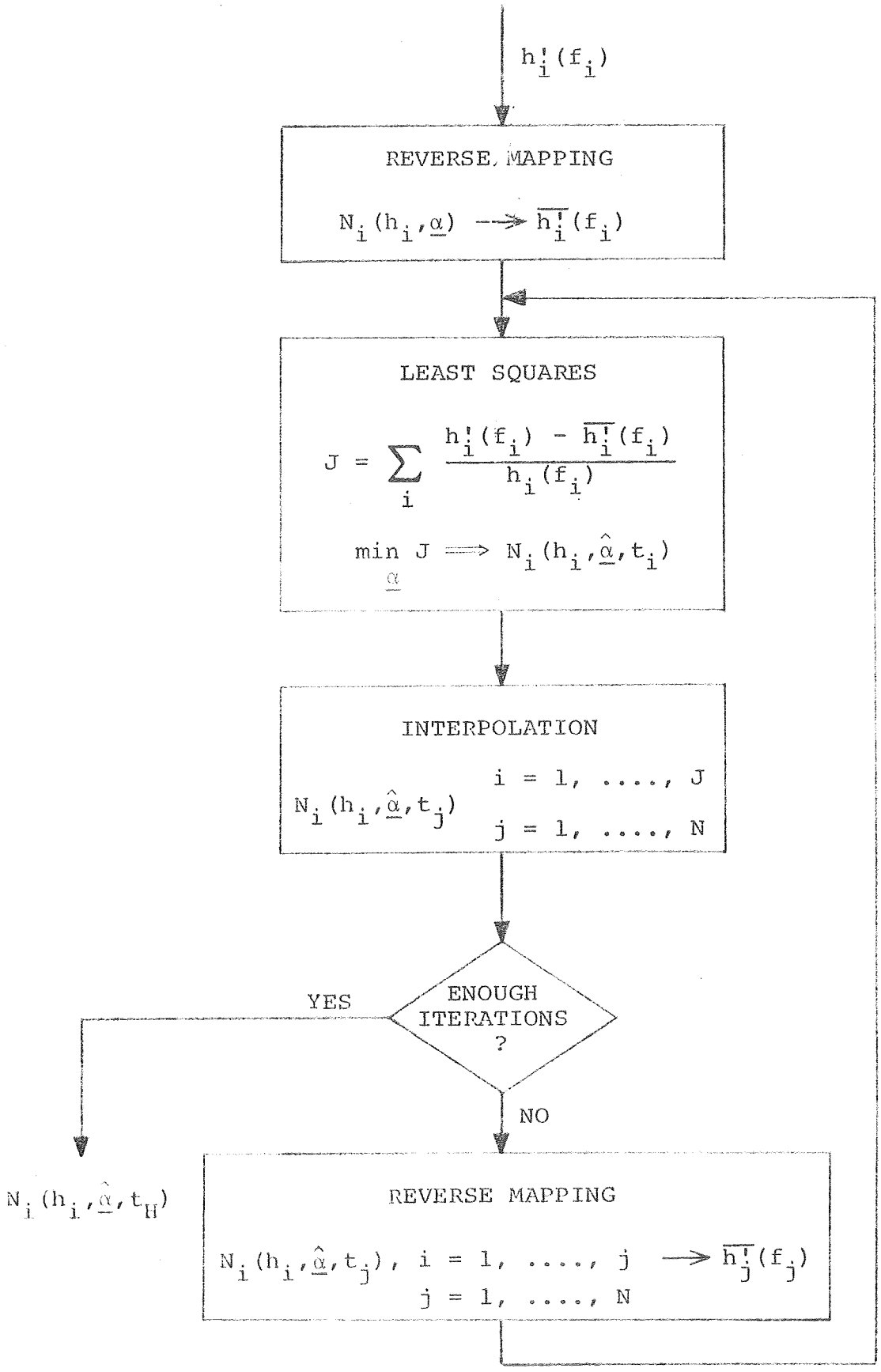


Fig. TR-7-5. Horizontal Processing Using Least Squares

Prepared for:

Astrodata, Inc.
240 E. Palais Rd.
Anaheim, Calif.

Astrodata Project 938400

TR-8

COMPUTATION OF THE PSEUDOINVERSE

February 16, 1970

John F. Kinkel
Technical Consultant
Newport Beach, Calif.

Abstract

A new computational algorithm is proposed for constructing the pseudoinverse of a matrix. The method is believed to be more efficient for machine computation than previous methods.

Introduction

A problem which arises frequently in the application of estimation theory[2] is to find a best estimate of $\underline{x}$ from

$$A\underline{x} = \underline{y} \tag{1}$$

where A is $m \times n$

$\underline{x}$ is $n \times 1$

$\underline{y}$ is $m \times 1$

If A is a square matrix of rank n then $\underline{x}$ is uniquely determined from

$$\underline{x} = A^{-1}\underline{y} \tag{2}$$

However, most frequently A is a rectangular matrix with $m > n$. For the case in which the rank of A is n, the least squares estimate can be found from[2]

$$\hat{\underline{x}} = (A^T A)^{-1} A^T \underline{y} \tag{3}$$

since $r(A) = n \;>\; r(A^T A) = n \;>\; (A^T A)$ is nonsingular

($r(A)$ = rank of A).

However, for cases in which $r(A) < n$ it follows that $r(A^T A) < n$ and $(A^T A)$ is singular so that eq. (3) does not apply. For these

cases the pseudoinverse[1-4] may be used. The pseudoinverse is defined by the following three requirements[4]:

$$\text{I} \quad A^{+}A\underline{x} = \underline{x} \qquad \forall \underline{x} \in N(A)^{\perp} = R(A^{T}) \tag{4}$$

$$\text{II} \quad A^{+}\underline{z} = \underline{0} \qquad \forall \underline{z} \in R(A)^{\perp} = N(A^{T}) \tag{5}$$

$$\text{III} \quad A^{+}(\underline{y} + \underline{z}) = A^{+}\underline{y} + A^{+}\underline{z} \qquad \forall \underline{y} \in R(A) = N(A^{T})^{\perp}, \quad \forall \underline{z} \in R(A)^{\perp} = N(A^{T}) \tag{6}$$

$$\text{where } \underline{y} \in N(A^{T})^{\perp}, \ \underline{z} \in N(A^{T}) \implies \underline{y}^{T}\underline{z} = 0 \tag{7}$$

$N(A)^{\perp}$ is the subspace in E_n which is the orthogonal complement of the null space of A,

$R(A^{T})$ is the restricted range space of A^{T} in E_n,

$R(A)^{\perp}$ is the subspace in E_m which is the orthogonal complement of the restricted range of A,

$N(A^{T})$ is the null space of A^{T} in E_m.

Therefore, we can decompose E_m and E_n as follows:

$$R(A) \subset E_m$$
$$N(A^{T}) \subset E_m$$
$$R(A^{T}) \subset E_n$$
$$N(A) \subset E_n$$

$$E_m = N(A^T)^{\perp} \oplus N(A^T)$$
$$= R(A) \oplus R(A)^{\perp}$$
$$R(A) = N(A^T)^{\perp}$$
$$R(A)^{\perp} = N(A^T)$$

$$E_n = N(A)^{\perp} \oplus N(A)$$
$$= R(A^T) \oplus R(AT)^{\perp}$$
$$R(A^T) = N(A)^{\perp}$$
$$R(A^T)^{\perp} = N(A)$$

Notation

A	an m × n matrix
A^{-1}	inverse of A
A^{+}	pseudoinverse of A
A^{++}	pseudo-pseudoinverse of A (meets only requirement I)
A^{T}	transpose of A
$N(A)$	null space of A; $\underline{z} \in N(A) \Longleftrightarrow A\underline{z} = \underline{0}$
$R(A)$	a restricted range of A
$N(A)^{\perp}$	orthogonal complement of $N(A)$
E_m	space of all column vectors of m elements
E_n	space of all column vectors of n elements
$\forall$	for every
$\in$	an element of
$\ni$	such that
$\oplus$	direct sum of subspaces
$\subset$	contained in

For the case in which A is a nonsingular square matrix the pseudoinverse becomes the inverse. The pseudoinverse has the property that

$$\hat{\underline{x}} = A^{+}\underline{y} \tag{8}$$

yields an estimate for $\underline{x}$ which is best in the sense of least squares[1,2].

The computational methods described by Greville[1] and Deutsch[2] require the selection of linearly independent[6] sets of rows and columns in order to form the pseudoinverse. While this approach is very good for an understanding of the construction of the pseudoinverse and works well for simple contrived examples, it is awkward to apply in practice.

Aoki[3] overcomes this problem by developing a construction of the pseudoinverse using eigenvalues and normalized eigenvectors of $A^{T}A$ and AA^{T*}. While this method is straightforward and well suited for machine computation, the determination of eigenvectors is a lengthy process and unattractive from the standpoint of efficiency.

*Matrices are assumed real so that only simple transpose rather than conjugate transpose is used.

Proposed Method

Zadeh and Desoer[4] have shown that

$$A^{+} = (A^{T}A)^{+}A^{T} \tag{9}$$

so that the determination of the pseudoinverse of any rectangular matrix can be reduced to finding the pseudoinverse of a square symmetric matrix.

It is well known that any symmetric n^{th} order matrix has n linearly independent eigenvectors[7]. Therefore, according to Pease[9] $A^{T}A$ is semisimple and can be written in terms of its eigenvalues and projectors as

$$A^{T}A = \sum_{i=1}^{n} \lambda_i P_i \tag{10}$$

where λ_i = eigenvalues of $A^{T}A$

P_i = projectors of $A^{T}A$

We note that if $A^{T}A$ is singular then $\lambda_i = 0$ for some i, and that if $r(A^{T}A) = r$ there exists r nonzero values of λ_i. We define the set S as

$$S = \{i : \lambda_i \neq 0\} \tag{11}$$

Equation (10) can now be rewritten as

$$A^{T}A = \sum_{i \in S} \lambda_i P_i$$

Now we will show that the pseudoinverse of A^TA is given by

$$(A^TA)^+ = \sum_{i\epsilon S} \lambda_i^{-1} P_i \tag{12}$$

We see that

$$(A^TA)^+(A^TA) = \left(\sum_{i\epsilon S} \lambda_i P_i\right)\left(\sum_{i\epsilon S} \lambda_i^{-1} P_i\right) \tag{13}$$

Applying the fundamental property of projectors[9],

$$P_i P_j = \delta_{ij} P_i \tag{14}$$

to equation (13) gives

$$(A^TA)^+(A^TA) = \sum_{i\epsilon S} P_i \tag{15}$$

which satisfies the defining relations given in equations (4),(5),(6). Therefore, the pseudoinverse is given by eqn. (12). Next we shall develop a recursive method for finding the pseudoinverse.

Zadeh and Desoer[5] have described Fadeeva's method[8] for finding the resolvent $(sI - A^TA)^{-1}$ as follows:

$$(sI - A^TA)^{-1} = \frac{B(s)}{d(s)} = \sum_{i=1}^{n} \frac{P_i}{(s - \lambda_i)} \tag{16}$$

$$d(s) = s^n + d_1 s^{n-1} + d_2 s^{n-2} + ... + d_{n-1} s + d_n \tag{17}$$

$$B(s) = s^{n-1} B_0 + s^{n-2} B_1 + ... + s B_{n-2} + B_{n-1} \tag{18}$$

$$\begin{aligned} B_0 &= I & d_1 &= -\mathrm{tr}(A^T A) \\ B_1 &= B_0 A^T A + d_1 I & d_2 &= -1/2\ \mathrm{tr}(B_1 A^T A) \\ B_2 &= B_1 A^T A + d_2 I & d_3 &= -1/3\ \mathrm{tr}(B_2 A^T A) \\ B_k &= B_{k-1} A^T A + d_k I & d_k &= -1/k\ \mathrm{tr}(B_{k-1} A^T A) \\ B_{n-1} &= B_{n-2} A^T A + d_{n-1} I & d_n &= -1/n\ \mathrm{tr}(B_{n-1} A^T A) \\ \phi &= B_{n-1} A^T A + d_n I & & \text{where tr = trace} \end{aligned} \tag{19}$$

The inverse of a nonsingular matrix can be obtained from equations (16), (17), and (18) as

$$(A^T A)^{-1} = \frac{B(0)}{d(0)} = \frac{B_{n-1}}{d_n} \tag{20}$$

However, the pseudoinverse can be found more easily from eqn. (19).

First, suppose that

$$r(A^T A) = n \tag{21}$$

Since from equation (16) it is clear that the roots of d(s) are given by the λ_i, it follows that

$$d_n \neq 0 \tag{22}$$

Then from eqn. (19) we have

$$B_{n-1}A^TA + d_nI = \phi$$

and

$$(A^TA)^{-1} = -\frac{B_{n-1}}{d_n} \tag{23}$$

as before.

Now suppose that

$$r(A^TA) = n-1 > 0 \tag{24}$$

Then since one of the $\lambda_i = 0$ it follows that

$$d_n = 0 \tag{25}$$

$$d_{n-1} \neq 0$$

Using these results in eqn. (19) gives

$$B_{n-1}A^TA = \phi \tag{26}$$

$$(B_{n-2}A^TA + d_{n-1}I)A^TA = \phi \tag{27}$$

Premultiplying the last equation by A^TA gives

$$(A^TA\,\frac{B_{n-2}}{d_{n-1}}\,A^TA + A^TA)A^TA = \phi \tag{28}$$

For eqn. (28) to be satisfied

$$A^TA\,\frac{B_{n-2}}{d_{n-1}}\,A^TA + A^TA = \phi \tag{29}$$

The validity of the last statement may be shown as follows. From the way in which the B_i are formed in eqn (19) it is clear that B_i and A^TA commute. It is easily verified that matrices formed from sums and products of commuting symmetric matrices are symmetric*; therefore, from eqn. (19), the B_i are symmetric. If two real symmetric matrices commute, then there exists a common diagonalizing matrix, T, such that[10]

$$T^{-1}(A^TA)T = \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{bmatrix} \tag{30}$$

$$T^{-1}B_iT = \begin{bmatrix} v_1^{\,i} & & 0 \\ & \ddots & \\ 0 & & v_n^{\,i} \end{bmatrix} \tag{31}$$

Operating on eqn. (28) with T gives the following sequence

$$T^{-1}(A^TA\,\frac{B_{n-2}}{d_{n-1}}\,A^TA + A^TA)A^TAT = \phi \tag{32}$$

$$\{(T^{-1}A^TAT)(T^{-1}\,\frac{B_{n-2}}{d_{n-1}}\,T)(T^{-1}A^TAT) + T^{-1}A^TAT\}(T^{-1}A^TAT) = \phi \tag{33}$$

$$\left\{\begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{bmatrix}\begin{bmatrix} v_1^{\,n-2} & & 0 \\ & \ddots & \\ 0 & & v_n^{\,n-2} \end{bmatrix}\begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{bmatrix} + \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{bmatrix}\right\}\begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{bmatrix} = \phi \tag{34}$$

---

* $(AB)^T = B^TA^T = BA = AB$

$$(\lambda_i v_i^{n-2}\lambda_i + \lambda_i)\lambda_i = 0, \quad i = 1, 2, \ldots, n \tag{35}$$

But $r(A^TA) = n-1 > 0 \Longrightarrow \exists\ n-1 > 0$ values of $\lambda_i \neq 0$ so that we must have

$$\lambda_i v_i^{n-2}\lambda_i + \lambda_i = 0, \quad i = 1, 2, \ldots, n \tag{36}$$

$$< \quad > \quad A^TA \frac{B_{n-2}}{d_{n-1}} A^TA + A^TA = \phi$$

Therefore, eqn. (29) must hold.

The requirement of eqn. (4) can be restated by premultiplying both sides by A to give

$$AA^+A\underline{x} = A\underline{x} \qquad \forall \underline{x} \in N(A)^{\perp} = R(A^T) \tag{37}$$

Let us denote by A^{++} a matrix which meets requirement (4) but not necessarily (5) and (6). Comparison of eqn. (29) with the form of eqn. (4) given above gives

$$(A^TA)^{++} = -\frac{B_{n-2}}{d_{n-1}} \tag{38}$$

We could call A^{++} the pseudo-pseudoinverse of A since it satisfies only requirement I equation (4) but not necessarily requirements II and III, eqn. (5) and eqn. (6).

Next we consider the case for which

$$r(A^TA) = n-2 > 0 \tag{39}$$

Then since two of the $\lambda_i = 0$ it follows that

$$\begin{aligned} d_n &= 0 \\ d_{n-1} &= 0 \\ d_{n-2} &\neq 0 \end{aligned} \tag{40}$$

Using these results in eqn. (19) gives

$$B_{n-1}A^TA = \phi \tag{41}$$

$$B_{n-2}(A^TA)^2 = \phi \tag{42}$$

$$B_{n-2} = B_{n-3}A^TA + d_{n-2}I \text{ from (19)}$$

$$(B_{n-3}(A^TA) + d_{n-2}I)(A^TA)^2 = \phi \tag{43}$$

Premultiplying both sides by A^TA gives

$$(A^TA\,\frac{B_{n-3}}{d_{n-2}}\,A^TA + A^TA)(A^TA)^2 = \phi \tag{44}$$

For eqn. (44) to be satisfied

$$(A^TA\,\frac{B_{n-3}}{d_{n-2}}\,A^TA + A^TA) = \phi \tag{45}$$

which can be shown by a diagonalizing transformation as in the case for $r(A^TA) = n-1 > 0$.

Therefore, eqn. (45) must hold so that

$$(A^TA)^{++} = \frac{B_{n-3}}{d_{n-2}} \tag{46}$$

Continuing in this manner we find that for

$$r(A^TA) = n-j > 0 \tag{47}$$

we have

$$(A^TA)^{++} = -\frac{B_{n-1-j}}{D_{n-j}} \tag{48}$$

It is not difficult to find examples of matrices such that*

$$(A^TA)^{++} \neq (A^TA)^{+} \tag{49}$$

However, we can show that

$$A^{+} = (A^TA)^{++}A^T \tag{50}$$

is a valid extended version of eqn. (9). First consider requirement I of eqn. (4) in the following form

$$\text{I} \quad AA^{+}A\underline{x} = A\underline{x} \qquad \forall \underline{x} \in N(A)^{\perp}$$

---

*A simple example suggested by C. W. Barnes is $A = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ for which $(A^TA)^{++} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ and $A^{+} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$

Inserting our proposed solution, given above, yields

$$A[(A^TA)^{++}A^T]A\underline{x} = A\underline{x} \tag{51}$$

$$A^TA(A^TA)^{++}A^TA\underline{x} = A^TA\underline{x} \tag{52}$$

which is satisfied by the way in which A^{++} was defined.

Next we consider

$$\text{II} \quad A^+\underline{z} = \underline{0} \qquad \forall \underline{z} \in N(A^T)$$

Inserting our proposed expression for A^+ gives

$$(A^TA)^{++}A^T\underline{z} = 0 \qquad \forall \underline{z} \Rightarrow A^T\underline{z} = \underline{0}$$

so that requirement II is satisfied.

Finally we consider

$$\text{III} \quad A^+(\underline{y} + \underline{z}) = A^+\underline{y} + A^+\underline{z} \qquad \forall \underline{y} \in N(A^T)^{\perp}, \quad \forall \underline{z} \in N(A^T)$$

Substituting our proposed solution gives

$$(A^TA)^{++}A^T(\underline{y} + \underline{z}) = (A^TA)^{++}A^T\underline{y} + (A^TA)^{++}A^T\underline{z}$$

so that our proposed solution results in the proper decomposition of the space as given by the direct sum

$$E_N = N(A^T)^{\perp} + N(A^T)$$

Therefore we have shown that

$$A^{+} = (A^{T}A)^{++}A^{T}$$

where $(A^{T}A)^{++}$ meets only requirement I (eqn. (4)).

Combining this last result with eqn. (48) gives

$$A^{+} = -\frac{B_{n-1-j}}{d_{n-j}} A^{T} \tag{53}$$

However, since we are not directly interested in the rank of $A^{T}A$ we can find A^{+} as

$$A^{+} = -\frac{B_{k-1}}{d_{k}} A^{T} \tag{54}$$

where k is the largest value of i for which $d_i \neq 0$.

We see from eqn. (19) that approximately n^3 scalar multiplications are required to evaluate B_i and d_i, i = 1, 2, ..., n. Therefore, the number of computations rquired to compute the pseudoinverse by this method is the same order of magnitude as required to invert a nonsingular matrix by the Gauss-Jordan method.

The steps in the computation of the pseudoinverse of A may be summarized as follows:

1. Form $A^T A$
2. Compute B_i, d_i, i = 1, 2, ..., n from eqn. (19)
3. Find k as the largest i for which $d_i \neq 0$
4. Compute the pseudoinverse of A from eqn. (54)

Example Problem

The algorithm described by this paper for finding the pseudo-inverse has been programmed in BASIC as shown by the flow diagram of Fig. TR-8-1 and associated program listing. This program has been tested using the example problem given by Greville[1] with results as shown in Table TR-8-1. These results are in agreement with those obtained by Greville.

```
PSEUDØ       2-20-70
↑
LIST 10-490
10 REM PSEUDØINVERSE PRØGRAM
20 REM AX=Y
30 REM P IS PSEUDØINVERSE ØF A
40 REM X=PY IS LEAST SQUARES ESTIMATE
50 REM A HAS M RØWS, N CØLUMNS
60 REM ENTER "DATA (M),(N)" LINE NØ. 1000
70 REM ENTER MATRIX ELEMENTS STARTING WITH LINE NØ. 1001
80 REM NUMBER LINES CØNSECUTIVELY
90 REM BEGIN EACH LINE WITH "DATA"
100 REM SEPARATE ENTRIES BY CØMMAS
110 REM ENTER ELEMENTS IN FØLLØWING SEQUENCE:  A BY RØWS, Y
120 DIM A(10,10),X(10),Y(10),P(10,10),B(110,10),D(10),C(10,10)
130 REM B AND D ARE MATRICES IN FADEEVA'S METHØD
140 REM C=A(TRANSPØSE)*A
150 READ M,N
160 FØR I=1 TØ M
170 FØR J=1 TØ N
180 READ A(I,J)
190 NEXT J
200 NEXT I
210 FØR I=1 TØ M
220 READ Y(I)
230 NEXT I
240 REM FØRM C=A(TRANSPØSE)*A
250 FØR I=1 TØ N
260 FØR J=1 TØ N
270 LET C(I,J)=0.0
280 FØR K=1 TØ M
290 LET C(I,J)=C(I,J)+A(K,I)*A(K,J)
300 NEXT K
310 NEXT J
320 NEXT I
330 REM FØRM B(N*I+J,K), D(I) BY FADEEVA'S METHØD
340 LET D(1)=0.0
350 FØR I=1 TØ N
360 LET D(1)=D(1)-C(I,I)
370 NEXT I
380 FØR J=1 TØ N
390 FØR K=1 TØ N
400 LET B(N+J,K)=C(J,K)
410 NEXT K
420 NEXT J
430 FØR J=1 TØ N
440 LET B(N+J,J)=B(N+J,J)+D(1)
450 NEXT J
460 FØR I=2 TØ N
470 FØR J=1 TØ N
480 FØR K=1 TØ N
490 LET B(N*I+J,K)=0.0
>
```

```
 PSEUDØ       2-20-70       PAGE 2
*
LIST 500-1100
500 FØR L=1 TØ N
510 LET B(N*I+J,K)=B(N*I+J,K)+B(N*(I-1)+J,L)*C(L,K)
520 NEXT L
530 NEXT K
540 NEXT J
550 LET D(I)=0.0
560 FØR J=1 TØ N
570 LET D(I)=D(I)-B(N*I+J,J)
580 NEXT J
590 LET D(I)=D(I)/I
600 FØR J=1 TØ N
610 LET B(N*I+J,J)=B(N*I+J,J)+D(I)
620 NEXT J
630 NEXT I
640 REM FIND LARGEST I SUCH THAT D(I) UNEQUAL TØ ZERØ
650 FØR I=1 TØ N
660 IF ABS(D(N+1-I))<1.E-04 THEN 690
670 LET I1=N+1-I
680 GØTØ 700
690 NEXT I
700 REM CØMPUTE PSEUDØINVERSE
710 FØR I=1 TØ N
720 FØR J=1 TØ M
730 LET P(I,J)=0.0
740 FØR K=1 TØ N
750 LET P(I,J)=P(I,J)-B(N*(I1-1)+I,K)*A(J,K)/D(I1)
760 NEXT K
770 NEXT J
780 NEXT I
790 REM CØMPUTE X
800 FØR I=1 TØ N
810 LET X(I)=0.0
820 FØR J=1 TØ M
830 LET X(I)=X(I)+P(I,J)*Y(J)
840 NEXT J
850 NEXT I
860 PRINT "PSEUDØINVERSE IS"
870 PRINT
880 FØR I=1 TØ N
890 PRINT P(I,1),P(I,2),P(I,3)
900 NEXT I
910 PRINT
920 PRINT "I","X(I)"
930 FØR I=1 TØ N
940 PRINT I,X(I)
950 NEXT I
960 PRINT
970 PRINT "RANK ØF A=";I1
1100 END
  >
```

```
 LIST 1000-1004
1000 DATA 3,4
1001 DATA 4,-1,-3,2
1002 DATA -2,5,-1,-3
1003 DATA 2,13,-9,-5
1004 DATA 7,3,20
>

 RUN
PSEUDØINVERSE IS

 9.5029697E-02 -5.6580181E-02  2.031885E-02
-3.0790872E-02  3.3135355E-02  3.782432E-02
-6.7364802E-02  3.1884964E-02 -3.9074711E-02
 5.0640825E-02 -3.6730228E-02 -8.9090341E-03

I               X(I)
 1               .90184433
 2               .64035636
 3              -1.1573929
 4               6.6114411E-02

RANK ØF A= 2

>
```

Table TR-8-1

Conclusion

A computational algorithm using Fadeeva's method has been described for finding the pseudoinverse. The method is computationally efficient in that the number of operations required is of the same order as for the Gauss-Jordan method of matrix inversion.

Acknowledgements

The author wishes to thank Prof. David Isaacs (University of California, Irvine) for pointing out the work done by Greville[1].

I am particularly grateful to Prof. C. W. Barnes (University of California, Irvine) for pointing out the power of projectors in matrix analysis and for detecting an error in an earlier draft of this paper.

Bibliography

(1) Greville, T. N. E., "The Pseudoinverse of a Rectangular or Singular Matrix and its Application to the Solution of Systems of Linear Equations," SIAM Review, Vol. 1, No. 1, Jan. 1959, pp. 38-43.

(2) Deutsch, Ralph, Estimation Theory, Prentice-Hall, 1965, pp. 82-89.

(3) Aoki, Mansanao, Optimization of Stochastic System, Academic Press, 1967, pp. 318-324.

(4) Zadeh, Lofti A., Desoer, Charles A., Linear System Theory, McGraw-Hill, 1963, pp. 577-582.

(5) loc cit pp. 303-306

(6) Freedman, Bernard, Principles and Techniques of Applied Mathematics, Wiley, 1956.

(7) Murdoch, D. C., Linear Algebra for Undergraduates, Wiley, 1957.

(8) Fadeeva, V. N., Computational Methods in Linear Algebra, Dover, 1959.

[9] Pease, Marshall C., Methods of Matrix Algebra, Academic Press, 1965.

[10] Bellman, Richard, Introduction to Matrix Analysis, McGraw-Hill, 1960.

APPENDIX INTRODUCTION

The following eight Technical Reports were written by John Kinkel in response to an assignment to investigate the proposed methods of closed-loop ionospheric data reduction and provide the mathematical proofs necessary for a solid theoretical foundation.

TR-1 "Some Proposed Methods for Reduction of Topside Ionograms to Electron Density Profiles"

Methods are described for finding N(h) or N(h,t) from topside ionograms using approximation theory. A gradient technique is proposed to find the parameters of the approximating function.

TR-2 "A Weighted Least Squares Approximation Method"

The theory of Mallinckrodt's weighted least squares approximation method is investigated.

TR-3 "Inverse Mapping to Specified f"

A method is developed for computing values of $H(\bar{f},\underline{\alpha})$ at the same frequency as scaled data points in the h'(f) plane.

TR-4 "Simplification of Matrix Inversion Problem"

A method is developed for solving for $\underline{X}$ in the matrix equation $A\underline{X} = \underline{B}$, without computing A^{-1}.

TR-5 "Time Skew Problem"

The time skew problem in Horizontal Processing is investigated.

TR-6 "Optimal Step Size by One Dimensional Search"

A method is developed for optimizing the step size of a hyper-dimensional adjustment vector that is used in the matrix method.

TR-7 "Horizontal Processing"

Some of the considerations in Horizontal Processing are investigated.

TR-8 "Computation of the Pseudoinverse"

The development and proof of a new computational algorithm for constructing the pseudoinverse of a matrix is given.